Manuel d'utilisation de FAI (Fully Automatic Installation)

Thomas, Lange

Traduction française: Daniel, Leprince

Version 2.4.3 pour FAI version 2.6

Copyright © 2000, 2004 Thomas Lange

Licence

Ce produit est placé sous licence libre; vous pouvez le redistribuer et/ou le modifier conformément à la licence GNU General Public License (GPL), publié par la Free Software Foundation (FSF), dans la version 2, ou (à votre convenance) n'importe quelle version ultérieure.

Ce produit est distribué avec l'objectif d'être utile, mais sans aucune garantie, sans même la garantie de valeur commerciale ou d'aptitude à un but particulier. Voir la licence GNU-GPL pour plus de détails.

Une copie de la licence GNU-GPL est disponible dans le fichier `/usr/share/common-licenses/GPL` de la distribution *Debian GNU/Linux* ou sur le site Internet de GNU (http://www.gnu.org/copyleft/gpl.html [http://www.gnu.org/copyleft/gpl.html]).

Vous pouvez aussi l'obtenir en écrivant à

Free Software
Foundation, Inc., 59 Temple Place - Suite 330, Boston, MA 02111-1307,
USA

.

11 août 2004

Résumé

Ce guide décrit l'outil FAI, qui permet l'installation entièrement automatique des postes sous *Debian GNU/Linux*.

Cela inclut l'installation du logiciel, la préparation et la configuration des installations, et comment traiter les erreurs.

Table des matières

Introduction 2
- Sources 2
- Origine et objectifs de FAI 3
- Vue d'ensemble et concepts 3
- Comment fonctionne FAI ? 4
- Fonctions 4

Installation FAI 5
- Prérequis 5
- Comment créer un miroir Debian local 6
- Installation de FAI 6

Préparation du démarrage (boot) 9

Démarrage avec un boot PROM sur une carte 3Com 9
Démarrage avec une carte réseau respectant la norme PXE 9
Création d'une disquette de boot 10
Démarrage à partir d'un CD-ROM 10
Collecte des adresses Ethernet 10
Configuration du démon BOOTP 11
Configuration du démon DHCP 12
Messages de boot 12
Collecte d'informations système complémentaires 14
Vérification des paramètres des serveurs BOOTP et DHCP 15
Redémarrage de l'ordinateur 15
Vue d'ensemble de la séquence d'installation 15
Surveillance de l'installation 16
Configuration FAI 16
Définition des classes, variables et chargement des modules du noyau 16
Partitionnement de disques locaux, création de systèmes de fichiers 17
Installation des logiciels 17
Configuration spécifique de site 17
Sauvegarde des fichiers de log 18
Redémarrer le nouveau système installé 18
Pour l'utilisateur impatient 18
Préparez votre installation, FAI suivra vos instructions 19
Détails de l'installation 19
L'espace de configuration 20
Les tâches par défaut 21
Les routines de paramétrage des clients 22
Le concept de classe 23
Définition de classes 24
Définition de Variables 25
Configuration du disque dur 26
Configuration des paquets logiciels 26
Scripts dans /fai/scripts 27
Changer l'unité de démarrage 27
Crochets (Hooks) 28
Recherche les erreurs 29
Comment construire un cluster Beowulf en utilisant FAI 29
Préparation de la configuration Beowulf 29
Configurer le serveur maître 30
Outillage pour les clusters Beowulf 32
Wake on LAN avec les cartes réseau 3Com 32
FAI sur d'autres architectures 33
FAI sur AMD64 33
FAI sur PowerPC 33
FAI sur IA64 33
FAI pour Suse, Redhat et Gentoo 33
FAI sur du matériel SUN SPARC sous Linux 33
FAI pour Solaris 33
Utilisation de CVS avec FAI 34
Utiliser le contrôle de révision pour la configuration de FAI 34
Points divers 35
Fonctions utiles pour les administrateurs avancés 36

Introduction

Sources

La page d'accueil de FAI est http://www.informatik.uni-koeln.de/fai. Vous y trouverez toutes les informations sur FAI, par exemple les archives de la liste de diffusion. Le paquet FAI est aussi disponible comme un paquet *Debian* sur le site http://www.informatik.uni-koeln.de/fai/download. C'est un paquet *Debian* officiel et il est disponible sur tous les miroirs *Debian*. Pour accéder aux nouvelles versions de paquets FAI, ajoutez la ligne sui-

vante dans votre fichier `/etc/apt/sources.list`.

```
deb http://www.informatik.uni-koeln.de/fai/ download/
```

Envoyez vos rapports de bogue ou commentaires à `<fai@informatik.uni-koeln.de>`. Vous pouvez aussi utiliser le système de recherche de bogue de *Debian* (Bug Tracking System = BTS) http://www.debian.org/Bugs/ pour rapporter les d'erreurs.

Vous pouvez avoir accès au dépôt CVS contenant la dernière version en développement de FAI en utilisant les commandes suivantes en mode console. Le mot de passe d'établissement de la connexion est vide, appuyez simplement sur « **Entrée** ».

```
> CVSROOT=:pserver:anonymous@cvs.debian.org:/cvs/debian-boot
> cvs login
> cvs co -P fai-kernels
> cvs co -P fai
```

Vous pouvez aussi utiliser l'interface web du dépôt CVS à l'adresse : cvs.debian.org/fai/ [http://cvs.debian.org/fai/] (Et fai-kernels).

Origine et objectifs de FAI

Si vous êtes lassés d'exécuter des installations identiques d'un système d'exploitation de façon répétitive. Vous souhaitez être capable d'installer de façon simple un groupe de machines Linux identiques.

La répétition de la même tâche est ennuyeuse et source d'erreurs. Il est donc possible d'économiser beaucoup de temps en réalisant l'installation automatiquement.

En 1999, l'auteur, Thomas LANGE, a dû réaliser l'installation d'un cluster *Linux* avec un serveur maître et 16 clients. En se basant sur son expérience de l'installation automatique du système d'exploitation *Solaris*, sur du matériel *SUN SPARC*, l'idée de réaliser une installation automatique pour *Debian* est née. *Solaris* disposait d'une fonction d'installation automatique appelée *JumpStart*1. En s'appuyant sur les scripts d'auto-installation écrits par Casper Dik2, il a pu économiser beaucoup de temps, non seulement pour chaque nouvel ordinateur *SUN*, mais aussi pour la réinstallation des machines existantes. Par exemple, lors de la mise en place provisoire d'un réseau local avec quatre stations *SUN* pour une conférence, qui a duré seulement quelques jours. Il a pris ces stations dans le réseau de recherche normal et a préparé une installation spécifique pour la conférence. Lorsque tout a été fini, il a simplement réintégré les stations dans le réseau de recherche, redémarré une fois les machines et une demi-heure plus tard, tout fonctionnait comme auparavant. La configuration de tous les postes de travail était exactement la même qu'avant la conférence, parce que tout avait été exécuté suivant le même processus d'installation. La même procédure d'installation automatique est aussi utilisée pour réinstaller un poste de travail après le remplacement d'un disque dur endommagé. Il a fallu deux semaines pour recevoir le nouveau disque dur, mais seulement quelques minutes après la mise en place du nouveau disque pour que le poste de travail fonctionne comme auparavant. C'est ce qui l'a décidé à adapter cette technique à un cluster de PC sous *Linux*.12

Vue d'ensemble et concepts

FAI est un système non-interactif pour installer le système d'exploitation *Debian GNU/Linux* sans surveillance sur un ordinateur isolé ou sur un cluster complet. Vous pouvez prendre un ou plusieurs PC vierge, mettre sous tension et après seulement quelques minutes, *Linux* est installé, configuré et opérationnel sur tout le cluster, sans aucune interaction nécessaire. C'est aussi une méthode évolutive pour installer et mettre à jour un cluster sans surveillance et de façon simple. FAI utilise la distribution *Debian GNU/Linux* et un ensemble de scripts en *shell* et en *perl* dans la procédure d'installation. Les changements dans les fichiers de configuration du système d'exploitation peuvent être faits par *cfengine*, *shell*, *perl* ou d'autres langages de scripts.

FAI s'adresse aux administrateurs système qui doivent installer *Debian* sur un grand nombre d'ordinateurs. Parce que c'est un outil d'installation généraliste, il peut aussi être utilisé pour installer un cluster *Beowulf*, une ferme de calcul, un laboratoire *Linux* ou une salle de classe. Il est également facile de construire avec FAI des réseaux à grande échelle avec du matériel hétérogène ou des besoins d'installation différentes. Mais n'oubliez pas de préparer votre installation. Le chapitre Préparez votre installation, FAI suivra vos instructions fournit quelques conseils utiles sur ce sujet.

1Solaris 8™ *Advanced Installation Guide* sur docs.sun.com
2ftp://ftp.wins.uva.nl:/pub/solaris/auto.install/

Commençons par la description de quelques termes utilisés dans ce manuel.

Serveur d’installation	l'hôte sur lequel le paquet FAI est installé. Il fournit plusieurs services et les données pour toutes les machines « clientes » à installer. Dans les exemples de ce manuel cet hôte est appelé kueppers .
Client installé	une machine qui sera installée en utilisant FAI et une configuration préparée sur le serveur d'installation. Nous l'appellerons plus simplement *client* . Dans ce manuel, les clients exemples sont appelés *demohost* , *nucleus* , *atom01* , *atom02* ...
Configuration	Tous les détails sur la façon dont l'installation des clients doit être exécutée. Cela inclut les informations sur : • Le(s) disque(s) dur(s) • Les systèmes de fichiers locaux, leurs types, points de montages et options de montage • Les paquets de logiciels choisis • La configuration du clavier, du fuseau horaire, du NIS, de XFree86, du NFS, des comptes d'utilisateur, des imprimantes…
Nfsroot	C'est un système de fichiers (*chrooté*) placé sur le serveur d'installation. C'est le système de fichiers complet pour les clients pendant le processus d'installation. Tous les clients partagent le même *nfsroot*, qu'ils montent en lecture seule.

Comment fonctionne FAI ?

Le client qui va être installé en utilisant FAI, est démarré en boutant sur une disquette ou via la carte réseau. Il obtient alors une adresse IP et démarre un noyau *Linux,* puis monte son système de fichiers racine via NFS sur le serveur d'installation. Quand le système d'exploitation est opérationnel, le script de démarrage FAI exécute l'installation automatique, sans aucune interaction. D'abord, les disques durs sont partitionnés, les systèmes de fichiers sont créés et les paquets de logiciels sont installés. Ensuite, le nouveau système d'exploitation est configuré selon vos besoins locaux en utilisant quelques scripts. Finalement, le nouveau système d'exploitation va redémarrer en boutant sur le disque local.

Les détails sur la façon d'installer l'ordinateur (la configuration), sont stockés dans l'espace de configuration du serveur d'installation. Les fichiers de configuration sont partagés par les groupes d'ordinateurs qui ont une fonction similaire en utilisant le concept de classes. Donc vous n'avez pas besoin de créer une configuration pour chaque nouvelle machine. Il en ressort que FAI est une méthode évolutive pour l’installation d’un grand cluster avec un nombre important de noeuds.

FAI peut aussi être utilisé comme un système de dépannage en réseau. Vous pouvez démarrer votre ordinateur, mais sans exécuter d'installation. Au lieu de cela il démarrera un système *Debian GNU/Linux* complètement fonctionnel sans utiliser les disques durs locaux. Vous pouvez alors vous connecter à distance et sauvegarder ou restaurer une partition de disque, vérifier un système de fichiers, contrôler le matériel ou faire n’importe qu'elle autre tâche.

Fonctions

- Exécution d’une installation entièrement automatisée.
- Installation sans surveillance très rapide
- Les hôtes peuvent bouter sur la disquette ou sur la carte réseau

- Création facile de la disquette de boot (commune).
- Les protocoles DHCP et BOOTP sont supportés, ainsi que la méthode de boot PXE.
- Aucun disque virtuel initial n'est nécessaire, 8MO de RAM suffisent et celà fonctionne même sur une unité centrale à base de *i386*
- Le noyau d'installation peut utiliser des modules
- L'établissement d'une connexion via ssh est possible pendant le processus d'installation
- Deux terminaux virtuels complémentaires sont disponibles pendant l'installation
- Toute la configuration commune est partagée par tous les clients
- Les fichiers de log pour toutes les installations sont sauvegardés sur le serveur d'installation
- Les scripts *Shell*, *perl*, *expect* et *cfengine* sont supportés pour la mise en oeuvre de la configuration
- Accès à un miroir *Debian* via NFS, FTP ou HTTP
- Configuration du clavier paramétrable
- Peut être utilisé comme un système de secours
- Testé sur du matériel *SUN SPARC* sous *Linux* ou *Solaris*
- Système flexible grâce au concept de classes
- Classes *Beowulf* prédéfinies incluses
- Supporte les clients sans disque
- Ajout facile de fonctions personnalisées via un système de crochets (NdT : « *hooks* »)
- Changement facile du comportement par défaut via des crochets
- Support de LILO et GRUB
- Support des système de fichiers ReiserFS, ext3 et XFS
- Détection de matériel automatique
- Le démarrage et l'installation à partir d'un *CD-ROM* sont en préparation.

Installation FAI

Prérequis

Les points suivants sont nécessaires pour une installation avec FAI.

Un ordinateur	l'ordinateur doit avoir une carte d'interface réseau. Un disque dur local est aussi nécessaire, sauf pour une installation sans disque. Aucune disquette ni *CD-ROM* n'est nécessaire. Le clavier ou la carte graphique ne sont pas obligatoires.
Un serveur DHCP ou BOOTP	les clients ont besoin d'un de ces démons pour obtenir l'information de boot. Mais vous pouvez aussi mettre toute cette information sur la disquette de boot.
Un serveur TFTP	le démon TFTP est utilisé pour transférer le noyau aux clients. C'est nécessaire uniquement en boutant à l'aide d'une carte réseau disposant d'un boot PROM.

Point de montage racine client c'est un répertoire pouvant être monté, qui contient le système de fichiers complet pour les clients, pendant l'installation. Il sera créé pendant l'installation du paquet FAI et est aussi appelé `nfsroot`.

Miroir Debian l'Accès à un miroir *Debian* est nécessaire. Un miroir local incluant tous les paquets *Debian* ou un *apt-proxy[(8)]* est recommandé si vous installez plusieurs ordinateurs.

Noyau d'installation une image de noyau qui supporte la carte réseau et monte son système de fichiers racine via NFS. Le paquet *Debian fai-kernels* fournit un noyau par défaut pour FAI.

Espace de configuration Cet ensemble de répertoires qui contiennent les données de configuration est monté via NFS par défaut. Mais vous pouvez aussi obtenir ce répertoire à partir d'un système de gestion de versions comme CVS.

Le démon TFTP et le serveur NFS seront activés automatiquement lors de l'installation du paquet FAI. Tous les clients doivent avoir une carte réseau qui est reconnue par le noyau d'installation.

Comment créer un miroir Debian local

Le script **mkdebmirror**[3] peut être utilisé pour créer votre propre miroir *Debian* local. Ce script utilise le script **debmirror**[4] et le protocole *rsync[(1)]*. Un miroir *Debian* 3.0 (*Woody*) partiel, pour l'architecture *i386*, sans les paquets source a besoin de 9.0GB d'espace disque. L'accès au miroir via NFS sera la méthode normale et la plus rapide dans la plupart des cas. Pour voir plus de messages lors de l'exécution du script, tapez **mkdebmirror --debug**. Vous n'avez pas besoin d'utiliser le compte *root* pour créer et maintenir le miroir *Debian*. Pour utiliser l'accès HTTP au miroir *Debian* local, il faut installer un serveur Web et créer un lien symbolique vers le répertoire local où vous avez placé le miroir :

```
# apt-get install apache
# ln -s /files/scratch/debmirror /var/www/debmirror
```

Créez un fichier `sources.list`[(5)] dans `/etc/fai` qui donne accès à votre miroir *Debian*. Un exemple est disponible dans `/usr/share/doc/fai/examples/etc`. Ajoutez aussi l'adresse IP du serveur *HTTP* à la variable `NFSROOT_ETC_HOSTS` dans `/etc/fai/make-fai-nfsroot.conf` si le client n'a pas d'accès au DNS.[3][4]

Installation de FAI

Avant l'installation de FAI, vous devez installer le paquet *fai-kernels*, qui contiennent les noyaux d'installation par défaut pour FAI. Vous pouvez installer les deux paquets en utilisant packageinstall.

Si vous voulez installer tous les paquets qui sont utiles pour FAI, utilisez la commande suivante :

```
# apt-get install mknbi dhcp3-server tftpd-hpa rsh-server wget syslinux
```

Vous pouvez aussi obtenir la dernière version de FAI et des *fai-kernels* sur la page de téléchargement de FAI et installer les paquets en utilisant la commande **dpkg**.

La configuration pour le paquet FAI (pas les données de configuration pour les clients) est définie dans `/etc/fai/fai.conf`. Les définitions qui ne sont utilisées que pour la création de la *nfsroot* se trouvent dans `/etc/fai/make-fai-nfsroot.conf`. Comme FAI n'utilise pas encore *debconf*, éditez ces fichiers avant d'exécuter **fai-setup**.

Les variables importantes suivantes sont définies dans `/etc/fai/make-fai-nfsroot.conf` :

FAI_DEBOOTSTRAP Pour construire le *nfsroot*, il y a une commande appelée **debootstrap**[(8)]. Cela nécessite l'adresse d'un miroir *Debian* et le nom de la distribution (*WOODY, SARGE, SID*) pour laquelle le système *Debian* doit être construit.

[3] Vous trouverez le script dans `/usr/share/doc/fai/examples/utils/`
[4] Disponible comme paquet *Debian* ou sur les site FAI

NFSROOT_ETC_HOSTS Si vous utilisez un accès *HTTP* ou *FTP* vers le miroir *Debian*, ajoutez son nom et son adresse IP à cette variable. Pour un noeud maître de cluster *Beowulf*, ajoutez-y les noms et adresses IP des deux réseaux. Cette variable n'est pas nécessaire si les clients ont accès à un serveur DNS.

FAI_SOURCES_LIST Cette variable est maintenant périmée. Utilisez le fichier `/etc/fai/sources.list` à la place.

KERNELPACKAGE Vous devez spécifier le paquet – construit avec **make-kpkg**[(8)] – qui inclut le noyau par défaut pour démarrer les clients. Le paquet *Debian fai-kernels* contient le noyau d'installation par défaut, qui supporte les protocoles DHCP et BOOTP.

NFSROOT_PACKAGES Cette variable contient une liste des paquets complémentaires qui seront ajoutés au *nfsroot*.

FAI_BOOT Pour quel(s) protocole(s), de DHCP et/ou BOOTP, le serveur doit-il créer des installations (quand **make-fai-nfsroot** est lancé). Par défaut, il doit créer l'installation pour les deux protocoles.

Les variables importantes suivantes sont définies dans `/etc/fai/fai.conf` :

FAI_LOCATION C'est le nom d'hôte et le répertoire distant de l'espace de configuration, qui sera monté via NFS. Sa valeur par défaut est `/usr/local/share/fai` mais certains préfèrent utiliser `/home/fai/config` ou `/var/fai/config`. Rappelez-vous que ce répertoire doit être exportée à tous les clients, pour que tous les fichiers puissent être lus par *root*.

FAI_DEBMIRROR Si vous avez un accès NFS à votre miroir local *Debian*, cette variable spécifie le système de fichiers distant. Il sera monté sur `$MNTPOINT`, qui doit aussi être défini. Ce n'est pas nécessaire si vous utilisez l'accès via FTP ou HTTP.

Le contenu de `/etc/fai/sources.list` et la variable `FAI_DEBMIRROR` sont utilisés par le serveur d'installation, mais aussi par les clients. Si votre serveur d'installation dispose de plusieurs cartes réseau avec des noms d'hôte différents pour chaque carte (comme dans le cas d'un serveur *Beowulf*), utilisez le nom du serveur d'installation connu par les clients.

FAI utilise **apt-get**[(8)] pour créer le système de fichiers *nfsroot* dans `/usr/lib/fai/nfsroot`. Il faut environ 230Mo d'espace libre sur le disque. Avant l'installation de FAI, vous aurez besoin du programme **imggen**5, si vous voulez démarrer sur une carte réseau *3Com*. Cet exécutable convertit des images *netboot* créées par **mknbi-linux**[(8)], il est alors possible de bouter sur les cartes réseau *3Com*. Mettez l'exécutable dans votre `path` (par exemple `/usr/local/bin`). Après avoir édité `/etc/fai/fai.conf` et `/etc/fai/make-fai-nfsroot.conf`, lancez **fai-setup**.

```
kueppers[~]# fai-setup
Adding system user fai...
Adding new user fai (101) with group nogroup.
Creating home directory /home/fai.
/home/fai/.rhosts created.
User account fai set up.
Creating FAI nfsroot can take a long time and will
need more than 230MB disk space in /usr/lib/fai/nfsroot.
Creating nfsroot for sarge using debootstrap
dpkg: base-passwd: dependency problems, but configuring anyway as you request:
 base-passwd depends on libc6 (>= 2.3.2.ds1-4); however:
  Package libc6 is not installed.
dpkg: base-files: dependency problems, but configuring anyway as you request:
.
.
.
Automatically converting /etc/network/interfaces succeeded.
Old interfaces file saved as interfaces.dpkg-old.
Creating base.tgz
`/etc/fai/sources.list' -> `etc/apt/sources.list'
Upgrading /usr/lib/fai/nfsroot
Adding additional packages to /usr/lib/fai/nfsroot:
module-init-tools dhcp3-client ssh file rdate hwinfo
bootpc rsync wget rsh-client less dump reiserfsprogs usbutils
dpkg-dev ext2resize hdparm smartmontools parted raidtools2 lvm2
dnsutils ntpdate dosfstools cfengine cvs jove
sysutils dialog discover mdetect libnet-perl netcat libapt-pkg-perl
grub lilo read-edid kudzu hwtools dmidecode
```

```
hostname: Host name lookup failure
/var/lib/dpkg/tmp.ci/preinst: line 7: lvscan: command not found
Creating SSH2 RSA key; this may take some time ...
Creating SSH2 DSA key; this may take some time ...
starting OpenBSD Secure Shell server: sshd.
`/etc/fai/fai.conf' -> `/usr/lib/fai/nfsroot/etc/fai/fai.conf'
`/etc/fai/make-fai-nfsroot.conf' -> `/usr/lib/fai/nfsroot/etc/fai/make-fai-nfsroot.conf'
`/etc/fai/sources.list' -> `/usr/lib/fai/nfsroot/etc/fai/sources.list'
DHCP environment prepared. If you want to use it, you have to enable the dhcpd and the tftp-hpa daemon.

Image Creator for MBA ROMs v1.01, Date: Nov 26, 2001
Design and Coding by Nick Kroupetski <NickKroupetski@hotmail.com>
Usage: imggen [OPTION] inputfile outputfile
  -a,   Add 3Com MBA/BootWare support
  -r,   Remove 3Com MBA/BootWare support from image file
  -i,   Show information on an image
  -h,   Help screen

In filename: /boot/fai/installimage
Out filename: /boot/fai/installimage_3com
Adding MBA support...
MBA support has been succesfully added
BOOTP environment prepared.
make-fai-nfsroot finished.                <= *
Stopping NFS kernel daemon: mountd nfsd.
Unexporting directories for NFS kernel daemon...done.
Exporting directories for NFS kernel daemon...done.
Starting NFS kernel daemon: nfsd mountd.
You have no FAI configuration. Copy the simple examples with:
cp -a /usr/share/doc/fai/examples/simple/* /usr/local/share/fai
Then change the configuration files to meet your local needs.
FAI setup finished.                       <= *
```

Un enregistrement complet de fai-setup est disponible sur le site de fai. *Il est important de voir les deux lignes marquées avec un astérisque*. Sinon, les choses se passeront mal. Si vous obtenez beaucoup de lignes blanches, il est probable que vous utilisez *konsole*, l'émulation terminal X pour KDE qui est boguée. Refaites alors un essai en utilisant *xterm*.

Les messages d'avertissement de *dpkg* sur des problèmes de dépendances peuvent être ignorés. Si vous avez des problèmes pendant **fai-setup**, ils proviennent en général de **make-fai-nfsroot**[(8)]. Vous pouvez le redémarrer en appelant '**make-fai-nfsroot -r**' (recover). Ajouter '`-v `' vous donne une production plus verbeuse qui peut vous aider à définir exactement l'erreur. Si vous voulez créer un fichier de log, vous pouvez utiliser

```
sudo /usr/sbin/make-fai-nfsroot -r -v 2>&1 | tee make-fai-nfsroot.log
```

Il peut être utile d'entrer manuellement dans l'environnement chrooté

```
sudo chroot /usr/lib/fai/nfsroot
```

La routine d'installation ajoute quelques lignes à `/etc/exports` pour exporter le *nfsroot* et l'espace de configuration à tous les hôtes qui appartiennent au groupe réseau *faiclients*. Si vous exportez déjà un répertoire parent de ces répertoires, vous pouvez commenter ces lignes, puisque le *nfs-kernel-server* à des problèmes pour exporter un répertoire et un de ses sous-répertoires avec des options différentes. Tous les clients doivent appartenir à ce groupe réseau pour monter ces répertoires avec succès. Les groupes réseau sont définis dans */etc/netgroup* ou dans la map NIS correspondante. Un exemple pour le fichier `netgroup` peut être trouvé dans `/usr/share/doc/fai/examples/etc/netgroup`. Pour plus d'information, lisez la page de manuel **netgroup**[(5)] et le *NIS HOWTO*. Après avoir changé les groupes réseau, le serveur NFS doit recharger sa configuration. Utilisez une des commandes suivantes, en fonction du serveur NFS que vous utilisez :

```
kueppers# /etc/init.d/nfs-kernel-server reload
kueppers# /etc/init.d/nfs-user-server reload
```

L'installation crée aussi le compte *fai* (défini par `$LOGUSER`) si il n'est pas déjà existant. Vous pouvez ajouter un utilisateur avant d'appeler **fai-setup**[(8)] en utilisant la commande **adduser**[(8)] et l'utiliser comme votre compte local pour sauvegarder les fichiers de log. Les fichiers de log de tous les clients sont sauvegardés dans le répertoire *home* de ce compte. Si vous boutez sur la carte réseau, vous devez changer le groupe primaire de ce compte, pour que ce compte ait la permission d'écrire dans `/boot/fai` de façon à changer les liens symboliques vers l'image du noyau qui est démarré par un client.

Après cela, FAI est installé avec succès sur votre serveur, mais n'a aucune configuration pour les clients. Inspirez-vous des exemples de `/usr/share/doc/fai/examples/simple/` en utilisant la commande **copy** et

lisez Détails de l'installation. Avant de pouvoir initialiser un démon DHCP ou BOOTP, vous devez rassembler des informations sur le réseau de tous vos clients. Reportez–vous à la section Création d'une disquette de boot.

Si vous faites des changements dans `/etc/fai/fai.conf`, `/etc/fai/make-fai-nfsroot.conf` ou si vous voulez installer un nouveau noyau dans *nfsroot*, l'arborescence *nfsroot* doit être reconstruite en lançant **make-fai-nfsroot**[(8)].[5]

Diagnostic des pannes lors de l'installation

L'installation de FAI ajoute le compte *fai*, exporte des systèmes de fichiers et appelle **make-fai-nfsroot**.

Si vous appelez **make-fai-nfsroot –v** (verbose), vous verrez plus de messages. En utilisant un miroir local *Debian*, il est important que le serveur d'installation puisse monter le répertoire via NFS. Si ce montage échoue, vérifiez `/etc/exports` et `/etc/netgroup`. Un exemple peut être trouvé dans `/usr/share/doc/fai/examples/etc/netgroup`.

Préparation du démarrage (boot)

Avant le premier démarrage, vous devez choisir quel moyen vous utiliserez pour bouter. Vous pouvez utiliser la disquette de boot ou configurer l'ordinateur pour démarrer via la carte réseau en utilisant un boot PROM, ce qui est beaucoup plus chic.

Démarrage avec un boot PROM sur une carte 3Com

Si vous avez une carte réseau *3Com* équipée d'un boot ROM par Lanworks Technologies ou qui inclut déjà le logiciel *DynamicAccess Managed PC Boot Agent* (MBA)[6], vous pouvez entrer dans le setup MBA en tapant Ctrl+Alt+B pendant le boot. Le setup devrait ressembler à ceci :

```
Managed PC Boot Agent (MBA) v4.00
(C) Copyright 1999 Lanworks Technologies Co. a subsidiary of 3Com Corporation
All rights reserved.
================================================================================
                            Configuration

Boot Method:                PXE

Default Boot:               Network
Local Boot:                 Enabled
Config Message:             Enabled
Message Timeout:            3 Seconds
Boot Failure Prompt:        Wait for timeout
================================================================================
  Use cursor keys to edit: Up/Down change field, Left/Right change value
  ESC to quit, F9 restore previous settings, F10 to save
```

Choisissez la méthode de boot PXE et autorisez le boot local dans ce menu. Ainsi le premier équipement de boot sera la carte réseau en utilisant PXE, et le second doit être le disque dur local. Celà doit être configuré dans le BIOS de votre ordinateur. Si vous souhaitez utiliser le protocole BOOTP, sélectionnez TCP/IP, avec le protocole à BOOTP. En utilisant BOOTP, vous devez faire un lien symbolique du *hostname* de votre client vers l'image de noyau appropriée dans `/boot/fai`. Vous pouvez aussi utiliser l'utilitaire **tlink** (`/usr/share/doc/fai/examples/utils/tlink`) pour créer ce lien. Le fichier `installimage_3com` est créé par **imggen** et est prévu pour gérer le démarrage depuis les cartes réseau *3Com*[7].[6][7]

Démarrage avec une carte réseau respectant la norme PXE

Les cartes réseau les plus modernes supportent l'environnement de démarrage en boot PXE. Quelques cartes réseau (par exemple *Intel EtherExpress PRO 100*) ont une configuration de boot fixe, elles ne peuvent donc utiliser que le protocole de boot PXE. Cela exige un chargeur de boot PXE *Linux* et une version spéciale du démon TFTP, qui est disponible dans le paquet *Debian tftpd-hpa* . Installez d'abord les paquets nécessaires suivants :

[5]Disponible sur la page de téléchargement de http://www.ltsp.org ou sur http://www.informatik.uni-koeln.de/fai/download.
[6]http://support/3com.com/infodeli/tools/nic/mba.htm
[7]Si vous avez des problèmes en boutant avec une carte réseau 3com™ (message d'erreur «BOOTP record too large» quand le noyau vient d'être chargé sur le PC) essayez le programme **imggen-1.00** pour convertir l'image *netboot* en image *installimage_3com*. Ce problème a été relevé en utilisant *netboot 0.8.1-4* et *Image Creator for MBA ROMs V1.01*, mais seulement avec un processeur *Athlon*.

```
# apt-get install dhcp3-server syslinux tftpd-hpa
```

Configurez alors le démon DHCP. Des fichiers de configuration types peuvent être trouvés dans `/usr/share/doc/fai/examples/etc/dhcpd.conf` . Copiez ce fichier dans `/etc/dhcp3/dhcpd.conf` . Validez ensuite le démon *tftp* spécial en utilisant cette ligne dans le fichier `/etc/inetd.conf` :

```
tftp dgram udp wait root /usr/sbin/in.tftpd in.tftpd -s /boot/fai
```

Le client charge le bootloader pxelinux qui reçoit sa configuration via TFTP depuis un fichier dans le répertoire /boot/fai/pxelinux.cfg. En utilisant la commande **fai-chboot**[(8)], vous pouvez choisir quel noyau sera chargé par le chargeur PXE *Linux* etquels paramètres aditionnels sont passés à ce noyau. Lisez les pages de manuels, qui vous donnent aussi quelques bons exemples.Voir `/usr/share/doc/syslinux/pxelinux.doc` pour avoir plus d'information sur ce genre d'environnements de boot. L'environnement PXE utilise l'image originale du noyau (pas l'image de *netboot* faite par **mknbi-linux**) qui est copié dans `/boot/fai/vmlinuz-install`.

Création d'une disquette de boot

Si votre carte réseau ne permet pas le boot PXE, vous avez deux options. La première est de créer une disquette de boot qui utilise **etherboot**, vous pouvez alors utiliser DHCP et TFTP pour obtenir le noyau d'installation qui a été créé avec **mknbi-linux**[(8)]. Beaucoup de cartes éthernet supportent le démarrage via éthernet avec une *EPROM* spéciale de boot ou en démarrant depuis une disquette (http://rom-o-matic.net/). Une documentation détaillée sur le démarrage via éthernet peut être consultée sur http://etherboot.sourceforge.net [http://etherboot.sourceforge.net/].

La deuxième option est de démarrer sur une disquette, créée avec la commande **make-fai-bootfloppy**[(8)]. Puisqu'elle ne contient aucune information spécifique au client, cette disquette convient pour démarrer l'installation de n'importe quel poste client. Vous pouvez aussi spécifier des paramètres complémentaires de noyau pour cette disquette de boot ou mettre d'autres variables, si nécessaire. Ne validez pas le support de BOOTP si vous avez un serveur DHCP en service sur votre réseau et vice versa. Cela peut provoquer des pertes d'informations. Consultez aussi la page de manuel pour **make-fai-bootfloppy**[(8)]. Si vous n'avez aucun serveur BOOTP ou DHCP, indiquez la configuration réseau en paramètres de lancement du noyau. Le format est :

```
ip=<client-ip>:<server-ip>:<gw-ip>:<netmask>:<hostname>:<device>:<autoconf>
```

Pour une information complémentaire, voir `/usr/src/linux/Documentation/nfsroot.txt` dans les sources du noyau.

Démarrage à partir d'un CD-ROM

Le travail est en cours pour créer un CD-ROM boutable qui permette le démarrage et l'installation d'un client. Nous expérons que celà deviendra prochainement un paquet officiel Debian. Il contiendra le *nfsroot*, l'espace de configuration et un sous-ensemble du miroir *Debian*, qui contient tous les logiciels dont vous avez besoin pour une installation sans surveillance. Regardez les archives de la *mailing-liste* de *FAI* pour plus d'information. C'est actuellement disponible sur http://holbytla.org/fai/.

Collecte des adresses Ethernet

Maintenant il est temps de commencer l'installation de vos clients. SI vous boutez directement maintenant, l'installation échouera sur le client, parce qu'aucun démon BOOTP ou DHCP ne s'exécute encore ou reconnaisse les hôtes. Mais vous pouvez utiliser cette première tentative de boot pour facilement rassembler toutes les adresses *Ethernet* des cartes réseau.

Vous devez relever toutes les adresses *Ethernet* (MAC) des clients et assigner un *hostname* et une adresse IP à chaque client. Une solution pour collecter toutes les adresses MAC est de démarrer tous vos clients. Pendant la phase de boot, les clients envoient des paquets de diffusion (*broadcast*) sur le réseau local. Vous pouvez enregistrer les adresses MAC de ces hôtes en exécutant la commande suivante sur le serveur pendant cette phase :

```
# tcpdump -qte broadcast and port bootpc >/tmp/mac.lis
```

Après l'envoi de quelques paquets de *broadcast* par les clients (le démarrage échouera parce que **bootpd** ne fonctionne pas ou ne reconnaît pas encore les adresses MAC) interrompre **tcpdump** en tapant ctrl-c . Vous obtiendrez une liste de toutes les adresses MAC uniques avec ces commandes :

```
# perl -ane 'print "\U$F[0]\n"' /tmp/mac.lis|sort|uniq
```

Après cela, vous devez simplement assigner ces adresses MAC à des noms machines (*hostname*) et des adresses IP (`/etc/ethers` et `/etc/hosts` ou les *maps* NIS correspondantes). Avec ces informations vous pouvez configurer votre démon BOOTP ou DHCP (voir la section « Configuration du démon BOOTP »en page suivante). Je recommande d'écrire les adresses MAC (derniers trois octets suffiront si vous avez des cartes réseau du même vendeur) et le *hostname* sur une étiquette devant chaque machine.

Configuration du démon BOOTP

Vous devrez utiliser cette méthode uniquement si vous ne pouvez pas utiliser de serveur DHCP. Il est en effet plus facile de créer et gérer la configuration pour DHCP. Vous trouverez une configuration d'exemple pour le démon BOOTP dans `/usr/share/doc/fai/examples/etc/bootptab`.

```
# /etc/bootptab example for FAI
# replace FAISERVER with the name of your install server

.faiglobal:\
 :ms=1024:\
 :hd=/boot/fai:\
 :hn:bs=auto:\
 :rp=/usr/lib/fai/nfsroot:

.failocal:\
 :tc=.faiglobal:\
 :sa=FAISERVER:\
 :ts=FAISERVER:\
 :sm=255.255.255.0:\
 :gw=134.95.9.254:\
 :dn=informatik.uni-koeln.de:\
 :ds=134.95.9.136,134.95.100.209,134.95.100.208,134.95.140.208:\
 :ys=rubens:yd=informatik4711.YP:\
 :nt=time.rrz.uni-koeln.de,time2.rrz.uni-koeln.de:

# now one entry for each install client
demohost:ha=0x00105A240012:bf=demohost:tc=.failocal:T172="verbose sshd createvt debug":
ant01:ha=0x00105A000000:bf=ant01:tc=.failocal:T172="sshd":
```

Insérez une ligne pour chaque client à la fin de ce fichier (comme pour les hôtes *demohost* et *ant01* dans le fichier exemple). Remplacez la chaîne *`FAISERVER`* par le nom de votre serveur d'installation. Si le serveur d'installation a plusieurs cartes réseau et noms d'hôte, utilisez le nom d'hôte de la carte réseau sur laquelle les clients se connectent. Ajustez ensuite les autres paramètres de réseau (*`sm`*, *`gw`*, *`dn`*, *`ds`*) à vos besoins locaux.

sm masque de sous-réseau

gw passerelle par défaut / routeur

dn nom de domaine

ds Liste de serveurs DNS. Le fichier `/etc/resolv.conf` sera créé en utilisant cette liste de serveurs DNS et le nom de domaine.

T172Liste des `FAI_FLAGS`; par exemple `verbose`, `debug`, `reboot`, `createvt`, `sshd`, `syslogd`.

Les paramètres pour NIS et les serveurs de temps (*`yp`*, *`yd`*, *`nt`*) sont facultatifs. Les paramètres avec le préfixe T (à partir de *`T170`*) sont des paramètres génériques, utilisés pour transférer quelques données spécifiques FAI aux clients[8].

Les Items de la liste `FAI_FLAGS` peuvent être séparés par un espace ou une virgule. `FAI_FLAGS` dans `bootptab` doit être séparé par un espace. Si vous définissez `FAI_FLAGS` comme des paramètres complémentaires du noyau, les éléments doivent être séparés par une virgule. Si vous n'avez pas le contrôle total du démon

BOOTP ou DHCP (parce que ce service est géré par une administration centralisée) vous pouvez aussi définir la variable FAI_ACTION dans des scripts /fai/class/*.var . Quand vous avez créé votre fichier bootptab , vous devez activer le démon BOOTP. Il est installé mais *Debian* ne l'active pas par défaut. Éditez /etc/inetd.conf et décommentez la ligne contenant #bootps. Rechargez alors la configuration de **inetd** .

```
# /etc/init.d/inetd reload
```

Le démon BOOTP recharge automatiquement le fichier de configuration si n'importe quel changement y est fait. Le démon pour DHCP doit toujours être redémarré manuellement après un changement dans le fichier de configuration. Maintenant il faut redémarrer tous les clients! FAI peut exécuter plusieurs actions au démarrage du client. Ces actions sont définies dans la variable FAI_ACTION. Soyez très prudent si vous positionnez FAI_ACTION sur *install* . Cela peut détruire toutes les données sur le client, car c'est la plupart de temps ce qu'il est supposé faire ;-). Il est donc recommandé de changer ce paramètre uniquement dans la base par client dans la configuration BOOTP. Ne le changez pas dans la section *.failocal* dans /etc/bootptab , qui est la définition pour l'ensemble des clients.[8]

Résolution de problèmes sur le démon BOOTP

Le démon BOOTP peut aussi être démarré en mode debug si ce n'est pas activé dans inetd.conf :

```
# bootpd -d7
```

Configuration du démon DHCP

Un exemple pour dhcp.conf[(5)] est fourni dans /usr/share/doc/fai/examples/etc , qui fonctionne avec la version 3.x du démon DHCP. Commencez en utilisant cet exemple et observez toutes les options utilisées. Si vous faites des changements à cette configuration, vous devez redémarrer le démon.

```
# /etc/init.d/dhcp3-server restart
```

Donc il est recommandé de mettre uniquement des données qui changent rarement dans ce fichier de configuration. Par défaut, le démon DHCP écrit son fichier journal dans /var/log/daemon.log . La commande **faichboot**[(5)] est utilisée pour créer une configuration par hôte pour l'environnement *pxelinux* .

Messages de boot

Voici les messages affichés lors du boot sur la disquette.

```
GRUB loading stage2..............
< now the grub menu with multiple boot options is displayed >
BOOTING 'FAI-BOTH'
kernel (fd0)/vmlinuz-2.4.26 root=/dev/nfs ip=both
  [Linux-bzImage, setup=0x1400, size=0xd8450]

Uncompressing Linux... OK, booting the Kernel.
Linux version 2.4.26 (root@kueppers) (gcc version 2.95.4 20011002
.
.
.
```

Après cela, les messages suivants seront identiques à ceux obtenus en boutant par la carte réseau. En boutant sur la carte réseau avec PXE vous verrez :

```
Managed PC Boot Agent (MBA) v4.00
.
.
Pre-boot eXecution Environment (PXE) v2.00
.
.
DHCP MAC ADDR: 00 04 75 74 A2 43
DHCP.../
CLIENT IP: 192.168.1.12 MASK: 255.255.255.0  DHCP IP: 192.168.1.250
GATEWAY IP: 192.168.1.254
```

[8] *T170*=FAI_LOCATION (maintenant définie dans fai.conf) et *T171*=FAI_ACTION. Vous pouvez définir ces variables dans le script class/*.var. Mais pour assurer une compatibilité ascendante, vous pouvez aussi définir ces variables depuis un server BOOTP ou DHCP.

```
PXELINUX 2.04 (Debian, 2004-06-24)  Copyright (C) 1994-2003 H. Peter Anvin


UNDI data segment at:   0009D740
UNDI data segment size: 3284
UNDI code segment at:   00090000
UNDI code segment size: 24C0
PXE entry point found (we hope) at 9D74:00F6
My Ip address seems to be C0A801C0 192.168.1.12
ip=192.168.1.150:192.168.1.250:192.168.1.254:255.255.255.0
TFTP prefix:
Trying to load pxelinnux.cfg/00-04-75-74-A2-43
Trying to load pxelinnux.cfg/C0A801C0
Loading vmlinuz-install.....................Ready.


Uncompressing Linux... OK, booting the Kernel.
Linux version 2.4.26 (root@kueppers) (gcc version 2.95.4 20011002
.
.
.
Sending DHCP requests ., OK
IP-Config: Got DHCP answer from 192.168.1.250, my address is 192.168.1.12
IP-Config: Complete:
   device=eth0, addr=192.168.1.12, mask=255.255.255.0, gw=192.168.1.254,
   host=demohost, domain=localdomain, nis-domain=(none),
   bootserver=192.168.1.250, rootserver=192.168.1.250, rootpath=/usr/lib/fai/nfsroot,rw,rsize=8192,wsize=
Looking up port of RPC 1000003/2 on 192.168.1.250
Looking up port of RPC 1000005/1 on 192.168.1.250
VFS: Mounted root (nfs filesystem).
.
.
             ------------------------------------------------------
               Fully Automatic Installation for Debian GNU/Linux
               FAI 2.6, 26 july 2004

               Thomas Lange      <lange@informatik.uni-koeln.de>
             ------------------------------------------------------


Calling task_confdir
Kernel parameters: ip=dhcp devfs=nomount FAI_ACTION=install root=/dev/nfs FAI_FLAGS=verbose,sshd,createvt
Defining variable: ip=dhcp
Defining variable: devfs=nomount
Defining variable: FAI_ACTION=install
Defining variable: root=/dev/nfs
Defining variable: FAI_FLAGS=verbose,sshd,createvt,syslogd
Defining variable: BOOT_IMAGE=vmlinuz-install
Reading /tmp/fai/boot.log
FAI_FLAGS: verbose=1
FAI_FLAGS: sshd=1
FAI_FLAGS: createvt=1
FAI_FLAGS: syslogd=1
Configuration space /fai mounted from faiserver:/usr/local/share/fai
Monitoring to server faiserver enabled.
Calling task_setup
.
.
Calling task_defclass
/usr/bin/fai-class: Defining classes.
.
.
Calling task_action
FAI_ACTION: install
Performing FAI installation. All data may be overwritten!
.
.
Disable swap device /dev/hda5
Sun Jul 25 22:10:26 CEST 2004
The installation took 296 seconds.
Calling task_chboot
Calling hook: savelog.LAST
ERRORS found in log files. See /tmp/fai/error.log.
savelog.LAST         OK.
Calling task_savelog
Save log files via rsh to fai@faiserver:demohost/install-20040725_220535
Calling task_faiend
Press <RETURN> to reboot or ctrl-c to execute a shell
```

Quand le message de copyright de FAI s'affiche, le client a monté le *nfsroot*9 comme répertoire racine (/) . C'est le système de fichiers complet pour le client à ce moment. Ensuite, task_confdir est exécuté. L'espace de

configuration est monté ou reçu d'un dépôt CVS.

Avant de commencer l'installation (FAI_ACTION=install), l'ordinateur émet trois bips sonores. Alors, soyez vigilant quand vous entendez trois signaux sonore, si vous ne souhaitez pas exécuter une installation![9]

Messages d'alerte au démarrage

Si votre carte réseau fonctionne, mais que le serveur d'installation n'exporte pas l'espace de configuration aux clients, le message suivant s'affichera :

```
Root-NFS: Server returned error -13 while mounting /usr/lib/fai/nfsroot
VFS: Unable to mount root fs via NFS, trying floppy.
VFS: Cannot open root device "nfs" or 02:00
Kernel panic: VFS Unable to mount root fs on 02:00
```

Utilisez la commande suivante pour voir quels répertoires sont exportés par le serveur d'installation (nommé *kueppers*) :

```
showmount -e kueppers
```

Le message d'erreur suivant indique que votre client n'obtient pas de réponse d'un serveur DHCP. Vérifiez vos câbles ou redémarrez le démon *dhcpd*[(8)] avec l'option debug activée.

```
PXE-E51: No DHCP or BOOTP offers received
Network boot aborted
```

Les messages suivants indiquent lorsque vous utilisez la méthode BOOTP qu'aucun serveur BOOTP ne répond.

```
Sending BOOTP requests ........ timed out!
IP-Config: Retrying forever (NFS root)...
```

Si vous obtenez le message d'erreur suivant, le pilote pour votre carte réseau n'est pas compilé dans le noyau d'installation.

```
IP-Config: No network devices available
Partition check:
 hda: hda1 hda2 < hda5 hda6 hda7 hda8 >
Root-NFS: No NFS server available, giving up.
VFS: Unable to mount root fs via NFS, trying floppy.
VFS: Insert root floppy and press ENTER
```

Vous devez alors compiler le pilote de votre carte réseau dans un nouveau noyau. Ce pilote ne doit pas être en module. Le fichier README du paquet fai-kernels /usr/share/doc/fai-kernels/README décrit comment compiler un noyau personnalisé. Ajustez ensuite la variable KERNELPACKAGE dans /etc/fai/make-fai-nfsroot.conf et reconstruisez la *nfsroot* à l'aide de la commande **make-fai-nfsroot**[(8)]. Ensuite, vous devez créer une nouvelle disquette si nécessaire. Maintenant, votre carte réseau est reconnue et le noyau peut monter la *nfsroot* avec succès.

Collecte d'informations système complémentaires

Maintenant que les clients ont démarré avec FAI_ACTION positionné à sysinfo . Tapez **Ctrl-c** pour obtenir une console ou utilisez **Alt-F2** ou **Alt-F3** pour changer de terminal, si vous avez ajouté createvt à FAI_FLAGS . La connexion à distance est disponible par un shell sécurisé si sshd est ajouté à FAI_FLAGS . Le mot de passe crypté est fixé par défaut à *fai* avec la variable FAI_ROOTPW de /etc/fai/make-fai-nfsroot.conf. Vous pouvez créer un mot de passe crypté en utilisant **mkpasswd**[(1)]. C'est le mot de passe *root* uniquement pendant le processus d'installation, pas pour le nouveau système installé . Vous pouvez aussi vous connecter sans mot de passe en utilisant SSH_IDENTITY . Pour vous connecter au client (nommé *demohost* dans cet exemple) depuis votre serveur d'installation, utilisez :

```
> ssh root@demohost
Warning: Permanently added 'demohost,134.95.9.200' to the list of known hosts.
root@demohost's password:
```

[9] /usr/lib/fai/nfsroot sur le serveur d'installation.

Vous avez maintenant un système *Linux* opérationnel sur le client, sans utiliser le disque dur local. Vous pouvez utiliser cette méthode comme système de secours si votre disque local est endommagé ou si l'ordinateur ne peut pas démarrer correctement à partir du disque dur. Vous obtiendrez une console ou vous pouvez exécuter des commandes diverses (**dmesg**, **lsmod**, **df**, **lspci**...). Regardez le fichier de log dans `/tmp/fai` . Vous y trouverez beaucoup d'informations sur le processus de boot. Tous les fichiers de log de `/tmp/fai` sont aussi copiés sur le serveur `$LOGSERVER` (si celui-ci n'est pas défini, sur le serveur d'installation) dans le répertoire `~fai/demohost/sysinfo/`[10]

Une fonctionnalité très utile de FAI est qu'il monte tous les systèmes de fichiers qu'il trouve sur les disques locaux, en lecture seulement. Il vous indique aussi sur quelle partition un fichier `/etc/fstab` est présent. Si une seule table de système de fichiers est trouvée, les partitions sont montées selon ces informations. Voici un exemple :

```
demohost:~# df
```

```
Filesystem 1k-blocks     Used Available Use% Mounted on
rootfs       2064192  1071184    888152  55% /
/dev/root    2064192  1071184    888152  55% /
shm            63548       76     63472   1% /tmp
kueppers:/usr/local/share/fai
             2064192   994480   964856   51% /fai
/dev/hda1      54447     9859    41777   19% /tmp/target
/dev/hda10   1153576       20  1141992    0% /tmp/target/files/install
/dev/hda9     711540       20   711520    0% /tmp/target/home
/dev/hda8     303336       13   300191    0% /tmp/target/tmp
/dev/hda7    1517948    98252  1342588    7% /tmp/target/usr
/dev/hda6     202225     8834   182949    5% /tmp/target/var
```

Cette méthode peut être utilisée comme environnement de secours! Dans l'avenir il sera possible de faire des sauvegardes ou restaurations de données sur les systèmes de fichiers existants. Si vous avez besoin d'un système de fichiers avec un accès en lecture-écriture, utilisez la commande **rwmount** :

```
demohost:~# rwmount /tmp/target/home
```

10

Vérification des paramètres des serveurs BOOTP et DHCP

Si le client démarre avec l'option sysinfo , vous pouvez aussi vérifier si toutes les informations des démons BOOTP ou DHCP sont reçues correctement. L'information reçue est écrite dans `/tmp/fai/boot.log` . Un exemple du résultat d'une requête DHCP peut être trouvé dans au paragraphe Les routines de paramétrage des clients.

Redémarrage de l'ordinateur

À tout moment vous pouvez redémarrer l'ordinateur en utilisant la commande **faireboot** , y compris à partir d'un terminal distant. Si l'installation n'est pas terminée, utilisez **faireboot -s** , alors les fichiers de log seront aussi copiés sur le serveur d'installation.

Vue d'ensemble de la séquence d'installation

Les tâches suivantes sont exécutées lors d'une installation, après le démarrage du noyau *Linux* sur les clients.

1. Mise en place de la configuration FAI
2. Chargement des modules de noyau

[10]Plus généralement : `~$LOGUSER/$HOSTNAME/$FAI_ACTION/`. Deux liens symboliques supplémentaires sont créés. Le lien symbolique *last* pointe sur le répertoire de log de la dernière action effectuée. Les liens symboliques *last-install* ou *last-sysinfo* pointent sus le répertoire de la dernière action correspondante. Des exemples de fichiers de log sont disponible sur la page d'accueil de FAI.

3. Définition des classes
4. Définition des variables
5. Partitionnement des disques locaux
6. Création et montage des systèmes de fichiers locaux
7. Installation des paquets de logiciels
8. Lancement des scripts de configuration spécifiques (de site)
9. Sauvegarde des fichiers de journal
10. Redémarrage du nouveau système installé

Vous pouvez aussi définir des programmes complémentaires ou des scripts qui seront lancés à des occasions particulières. Ils sont appelés *crochets*. Les crochets peuvent ajouter des fonctions complémentaires au processus d'installation ou remplacer les tâches secondaires par défaut de FAI. Il est alors très facile de personnaliser le processus d'installation complet. Les crochets sont expliqués en détail dans le chapitre Crochets (Hooks).

La durée de l'installation est déterminée par le nombre de logiciels, mais aussi par la vitesse du processeur et du disque dur. Voici quelques durées types. Tous les clients ont une carte réseau à 100Mbits/s. L'utilisation d'un réseau local à 10Mbits/s n'augmente pas considérablement le temps d'installation. Le réseau ne sera pas le goulot d'étranglement en cas d'installation simultanée de plusieurs PC clients.

- Athlon XP1600+ , 896MB,SCSI disk, 1 GB software 6 min
- AMD-K7, 500MHz , 320MB, IDE disk, 780 MB software 12 min
- PentiumPro 200MHz , 128MB, IDE disk, 800 MB software 28 min
- Pentium III 850MHz, 256MB, IDE disk, 820 MB software 10 min
- Pentium III 850MHz, 256MB, IDE disk, 180 MB software 3 min

Surveillance de l'installation

Vous pouvez contrôler l'installation des clients avec la commande **faimond**[(8)]. Tous les clients vérifient si le démon est lancé sur le serveur d'installation (ou sur la machine définie par la variable `monserver`). Alors, un message est envoyé quand une tâche commence et se termine. Le démon de surveillance *fai* affiche ce message sur la sortie standard. Une interface graphique sera développée dans le futur,.

Configuration FAI

Lors du démarrage du PC client, le script **/sbin/rcS_fai**[11] est exécuté. C'est le script principal qui contrôle la séquence des actions pour FAI. Aucun autre script dans `/etc/init.d/` n'est exécuté.

Un disque virtuel (*Ramdisk*) est créé et monté sur `/tmp` , qui est le seul répertoire accessible en écriture jusqu'à ce que les systèmes de fichiers locaux soient montés. Des paramètres complémentaires sont reçus du démon BOOTP ou DHCP et l'espace de configuration est monté via NFS depuis le serveur d'installation sur `/fai` . Le paramétrage est fini lorsque des terminaux virtuels complémentaires sont créés et que le démon de *shell* sécurisé (*ssh*) pour l'accès à distance est démarré.[11]

Définition des classes, variables et chargement des modules du noyau

[11]Depuis que la racine du système de fichier est montée par NFS, **rcS_fai** est situé dans `/usr/lib/fai/nfsroot/sbin` sur le serveur d'installation.

Le script **fai-class**[(1)] est utilisé pour définir les classes. Par suite, plusieurs scripts sont exécutés dans `/fai/class/` pour la définition des classes. Tous les scripts commençant par un chiffre **[0-9]*** sont exécutés par ordre alphanumérique. Chaque mot que ces scripts envoient sur la sortie standard sont interprétés comme des noms de classes. Les scripts se terminant par *.source* sont sourcés , ils peuvent ainsi définir de nouvelles classes en les ajoutant à la variable `newclasses` (voir **06hwdetect.source** pour un exemple). La sortie de ces scripts est ignorée. Ces classes sont définies pour le client en cours d'installation. On peut aussi dire que ce client appartient à ces classes. Une classe est définie ou indéfinie mais n'a pas de valeur. Seules les classes définies présentent un intérêt pour un PC client. La description de toutes les classes peut être trouvée dans `/usr/share/doc/fai/classes_description.txt`. Il est recommandé de documenter le travail réalisé par une nouvelle classe. Cette documentation sert de base pour composer l'ensemble de la configuration à partir des classes. Le script **06hwdetect.source** charge les modules nécessaires au noyau «à la demande». La description complète de tous ces scripts peut être trouvée dans Scripts dans /fai/scripts.

Le script **30menu.source** fait apparaître un petit menu qui demande à l'utilisateur quelle sorte d'installation doit être exécutée (par exemple : *Poste de travail CAO, portable, poste de travail scientifique, serveur de groupe de travail, poste bureautique* ...). Mais réfléchissez que cela ne mènera pas à une installation entièrement automatique ;-)

Après la définition des classes, chaque fichier `*.var` avec un préfixe qui correspond à une classe définie est exécuté pour définir les variables. Là, vous devez définir la variable `FAI_ACTION` et d'autres. Par défaut, `FAI_ACTION` est définie par la commande **fai-chboot**[(8)].

Partitionnement de disques locaux, création de systèmes de fichiers

Pour le partitionnement des disques, un fichier de configuration est choisi dans `/fai/disk_config` en utilisant les classes. C'est la description de la façon dont tous les disques locaux seront divisés, où les systèmes de fichiers seront créés (et leurs types comme *ext2*, *ext3*, *reiserfs*) et comment ils seront montés. Il est également possible de préserver le partitionnement du disque ou les données sur certaines partitions. C'est fait au moyen de la commande **setup_harddisks** , qui utilise **sfdisk** pour le partitionnement. Le format du fichier de configuration est décrit dans `/usr/share/doc/fai/README.disk_config`.

Pendant le processus d'installation, tous les systèmes de fichiers locaux sont monté relativement à `/tmp/target`. Par exemple `/tmp/target/home` deviendra `/home` dans le nouveau système installé.

Installation des logiciels

Quand les systèmes de fichiers locaux sont créés, ils sont vides (à l'exception des partitions préservées). Le système de base *Debian* et tous les logiciels demandés sont ensuite installés. D'abord les archives de base sont décompressées, puis la commande **install_packages**[(8)] installe tous les logiciels en utilisant **apt-get**[(8)] sans qu'aucune action manuelle ne soit nécessaire. Si un logiciel exige la présence d'un autre logiciel, **apt-get**[(8)] résout les dépendances et installe le paquet requis.

Les classes sont aussi utilisées lors de la sélection des fichiers de configuration dans `/fai/package_config/` pour l'installation des logiciels. Le format des fichiers de configuration est décrit dans Configuration des paquets logiciels.

Configuration spécifique de site

Une fois tous les logiciels installés, le système est presque prêt. Mais les configurations par défaut des logiciels ne répondront pas aux besoins spécifiques de votre site. C'est pourquoi vous pouvez lancer des scripts qui ajustent la configuration du système. Donc les scripts qui correspondent à un nom de classe dans `/fai/scripts` seront exécutées. Si `/fai/scripts/classname/` est un répertoire, tous les scripts nommés **S[0-9]** dans ce répertoire sont exécutés. Il est possible d'avoir plusieurs scripts de types différents (*shell*, *cfengine*...) à exécuter pour une même classe. FAI est fourni avec quelques exemples de scripts, mais vous pouvez écrire votre propre script (*bash*, *perl*, *cfengine*…).

Ces scripts importants sont décrits en détail dans Scripts dans /fai/scripts.

Sauvegarde des fichiers de log

Quand toutes les tâches d'installation sont terminées, les fichiers de journal sont sauvegardés dans `/var/log/fai/*$HOSTNAME*/install/`[12] sur le nouveau système et vers le compte sur le serveur d'installation si `$LOGUSER` est défini dans `/etc/fai/fai.conf` . Il est aussi possible de spécifier un autre hôte comme destination de la sauvegarde de log par un fichier dans `/fai/class/` . De plus, deux liens symboliques sont créés pour indiquer le dernier répertoire écrit. Cette méthode utilise **rsh**/**rcp** ou **ssh**/**scp** par défaut.

Vous pouvez utiliser d'autres méthodes de sauvegarde des logs sur le serveur distant. La méthode courante est définie par la variable `$FAI_LOGPROTO` dans le fichier `/etc/fai/fai.conf` :

ftp Cette option sauvegarde les journaux sur le serveur FTP distant défini par la variable `$LOGSERVER` (la valeur `$SERVER` est utilisée si non définie). La connexion au serveur FTP est faite en tant qu'utilisateur `$LOGUSER` , avec le mot de passe `$LOGPASSWD`. Le répertoire de log sur le serveur FTP est défini dans `$LOGREMOTEDIR`. Ces variables sont toutes définies dans le fichier `/etc/fai/fai.conf` . L'accès en écriture est nécessaire pour le `$LOGREMOTEDIR` sur le serveur FTP.

Tous les fichiers dans le répertoire `/tmp/fai` sont copiés sur le serveur FTP suivant cet exemple :

```
ftp://$LOGUSER:$LOGPASSWD@$LOGSERVER/$LOGREMOTEDIR/.
```

none Aucune sauvegarde des fichiers de log sur le serveur d'installation.

12

Redémarrer le nouveau système installé

Le système est automatiquement redémarré si «*reboot*» a été ajouté à `FAI_FLAGS` .En principe, le système nouvellement installé doit redémarrer depuis son second dispositif, le disque dur local. Pour sauter le boot sur la carte réseau, vous pouvez utiliser la commande **fai-chboot** pour valider le boot local.Si vous utilisez une disquette de boot, vous devez retirer la disquette sinon l'installation sera exécutée de nouveau. LisezChanger l’unité de démarrage pour savoir comment changer d'unité de lancement.

Pour l'utilisateur impatient

Bon ! Vous ne voulez pas lire le manuel en entier ? Vous préférez essayer l’installation sans lire le manuel ? OK. Voici comment le faire en quelques minutes .

- Installer fai et tous les logiciels recommandés (voir Installation FAI) sur votre serveur d’installation.
- Installer les exemples simples dans l'espace de configuration :

  ```
  cp -a /usr/share/doc/fai/examples/simple/* /usr/local/share/fai/
  ```

- Obtenir l'adresse MAC de votre hôte de test
- Ajouter votre hôte (essayez de le nommer `demohost`) dans `/etc/hosts` et `dhcpd.conf`
- Si votre `demohost` n'a aucun disque IDE, copiez le fichier `disk_config/SMALL_IDE` dans un fichier appelé `demohost` et remplacez le nom de disque `hda` par votre valeur spécifique.
- Si vous utilisez PXE, indiquez au client de charger le noyau d'installation et d'effectuez une installation au boot suivant.

  ```
  fai-chboot -IFv demohost
  ```

- Démarrer votre hôte de démonstration et profiter de l'installation complètement automatique.

12 `/var/log/fai/localhost/install/` est un lien vers ce répertoire.

- Si l'installation se termine avec succès, l'ordinateur doit démarrer un système Debian minimal . Vous pouvez ouvrir une session utilisateur « demo » avec le mot de passe « fai ».

Mais maintenant n'oubliez pas de lire le chapitre suivant Préparez votre installation, FAI suivra vos instructions !

Préparez votre installation, FAI suivra vos instructions

Avant de démarrer votre installation, vous devrez passer beaucoup de temps à sa préparation. Si votre plan d'installation est bien fait, FAI peut réaliser toutes les tâches ennuyeuses et répétitives pour transformer vos plans en réalité. FAI ne peut pas faire de bonnes installations si votre plan est imparfait ou s'il manque quelques détails importants. Commencez votre projet d'installation en répondant aux questions suivantes :

- Vais-je créer un cluster *Beowulf*, ou installer quelques machines de bureau ?
- A quoi ressemble ma topologie de réseau local ?
- Mon matériel est-il homogène ? Le matériel restera-t-il homogène dans l'avenir ?
- Le matériel a-t-il besoin d'un noyau spécifique ?
- Comment les machines seront-elles nommées (`hostnames`) ?
- Comment les disques durs locaux doivent-ils être partitionnés ?
- Quelles applications seront exécutées par les utilisateurs ?
- Les utilisateurs ont-ils besoin d'un système de file d'attente ?
- Quels logiciels doivent être installés ?
- Quels démons devront être lancés et comment seront-ils configurés ?
- Quels systèmes de fichiers distants devront être montés ?
- Comment les sauvegardes seront-elles exécutées ?
- L'alimentation électrique est-elle suffisante ?
- Quelle quantité de chaleur les noeuds de cluster produisent-ils et comment sont-ils refroidis ?
- ...

Vous devez aussi penser aux comptes utilisateurs, aux imprimantes, à la messagerie, à l'usage de *cron*, aux cartes graphiques, *dual boot*, *NIS*, *NTP*, *timezone*, à la configuration du clavier, à l'exportation et au montage des répertoires via NFS et à beaucoup d'autres choses. Il y a beaucoup à faire avant le démarrage d'une installation. Souvenez-vous que la connaissance est une force et qu'il ne tient qu'a vous de l'utiliser. L'installation et l'administration sont un procédé, pas un produit. FAI ne peut pas faire des choses que vous ne lui avez pas dit de faire.

Mais vous n'avez pas besoin de commencer à partir de zéro. Regardez tous les fichiers et les scripts dans l'espace de configuration. Il y a plein de choses que vous pouvez utiliser pour votre propre installation. Vous trouverez une bonne description qui présente plus d'aspects sur la construction d'une infrastructure sur : http://www.infrastructures.org/papers/bootstrap/ [http://www.infrastructures.org/papers/bootstrap/] *Bootstrap-*"*ping an Infrastructure*".

Détails de l'installation

L'espace de configuration

La configuration est l'ensemble des informations sur la façon précise d'installer un ordinateur. L'espace de configuration central pour tous les clients est placé sur le serveur d'installation dans `/usr/local/share/fai` et ses sous-répertoires. Ce répertoire sera monté par les clients sur `/fai`. Il est aussi possible de recevoir toutes les données de configuration à partir d'un dépôt cvs[(1)]. Les sous-répertoires suivants sont créés et incluent plusieurs fichiers :

class/ — Scripts et fichiers pour définir les classes et variables et pour charger des modules du noyau.

disk_config/ — Fichiers de configuration pour le partitionnement des disques et la création des systèmes de fichiers.

debconf/ — Ce répertoire rassemble toutes les données pour **debconf**[(8)]. Non encore utilisé !

package_config/ — Contient le fichier avec les listes de logiciels à installer ou supprimer.

scripts/ — Scripts pour la personnalisation locale (propre à chaque site).

files/ — Les fichiers utilisés par les scripts de personnalisation, par exemple les paquets de noyau (*kernel packages*) créés par l'utilisateur. La plupart des fichiers sont placés dans une arborescence de sous-répertoires qui reflète l'arborescence de répertoires standard. Par exemple, les modèles pour `nsswitch.conf` sont placés dans `/fai/files/etc/nsswich.conf` . Le répertoire `files/packages/` peut contenir vos paquets *Debian* locaux, qui peuvent être installés en les ajoutant aux variables `addpackages` . Voir Définition de Variables pour plus d'information.

hooks/ — Ce sont des programmes ou des scripts définis par l'utilisateur, qui sont lancés pendant le processus d'installation.
(NdT : nommés « crochets » dans le texte.)

Le script d'installation principal **rcS_fai** utilise tous ces sous-répertoires dans l'ordre listé, à part pour `hooks/` . Le paquet FAI contient des exemples pour tous ces scripts et fichiers de configuration dans `/usr/share/doc/fai/examples`. Copiez les exemples dans l'espace de configuration et commencez une installation. Ces fichiers n'ont pas besoin d'appartenir au compte *root*. Vous pouvez changer leur appartenance et éditer ensuite la configuration avec un compte utilisateur normal.

```
# cp -a /usr/share/doc/fai/examples/simple/* /usr/local/share/fai
# chown -R fai /usr/local/share/fai
```

Les fichiers suivants contiennent une configuration simple pour quelques exemples d'hôtes. Les exemples sont : un hôte de démonstration (appelé *demohost*) avec le bureau GNOME, et un cluster *Beowulf* avec un noeud maître appelé *nucleus* et des noeuds de calcul appelés *atom01* , *atom02* ....

demohost — Une machine avec un petit disque dur IDE installée avec GNOME et utilisant DHCP pour obtenir sa configuration réseau. C'est l'exemple le plus facile. Vous devez juste ajouter un fichier `XF86Config`.

nucleus — Ce noeud maître *Beowulf* est un serveur avec beaucoup de logiciels. Il fournit les répertoires `/home` et `/usr/local` pour ses noeuds de calcul. Quelques démons sont installés et activés par défaut.

atom01,... — Ces clients *Beowulf* montent `/usr/local` et `/home` sur *nucleus* . La majorité de l'espace disque est utilisé pour une partition vide, qui est exportée à un netgroup d'hôtes. Toutes les partitions vides sont montées sur tous les clients *Beowulf* via l'automonteur.

Commencez par regarder ces exemples et les étudier. Ensuite modifiez ou ajoutez des choses à ces exemples. Mais n'oubliez pas de préparer votre propre installation !

Les tâches par défaut

Après le démarrage du noyau, celui-ci monte le système de fichiers racine par NFS depuis le serveur d'installation et le processus **init**[(1)] démarre le script **/sbin/rcS_fai** . Ce script contrôle la séquence d'installation. Aucun autre script de `/etc/init.d/` n'est utilisé.

Le script d'installation utilise beaucoup de sous-programmes, qui sont définis dans `/usr/share/fai/subroutines` et un fichier spécifique au système d'exploitation13. Toutes les tâches importantes de l'installation sont appelées via le sous-programme **task** avec le nom de la tâche accolé comme une option (par exemple **task_instsoft**). Le sous-programme **task** appelle d'abord les *crochets* avec ce préfixe si il en existe et appelle ensuite la tâche par défaut (définie comme **task_name** dans les sous-programmes). La tâche par défaut et ses *crochets* peuvent être sautés sur demande en utilisant le sous-programme **skiptask()**.13

Ci-dessous la description de toutes les tâches par défaut.

confdir : le noyau a ajouté des variables définies par paramètres, le syslog et le démon de log du noyau sont démarrés. La liste des équipements réseau est stockée dans `$netdevices` . Des paramètres complémentaires sont alors récupérés d'un serveur DHCP ou BOOTP et des variables supplémentaires sont définies. Le fichier de configuration DNS est créé. L'espace de configuration est monté via NFS depuis le serveur d'installation sur `/fai` ou il est récupéré sur le dépôt CVS[(1)] correspondant. Pour utiliser un dépôt CVS, vous devez positionner les variables `$FAI_CVSROOT` , `$FAI_CVSTAG` , `$FAI_CVSMODULE` . Pour les détails, regardez le sous-programme get_fai_cvs() . Après cela, le fichier `/fai/hooks/subroutines` est sourcé si il existe. En utilisant ce fichier, vous pouvez définir vos propres sous-programmes ou ignorer la définition des sous-programmes de FAI.

setup : Cette tâche positionne l'horloge système, tous les `FAI_FLAGS` sont définis et deux terminaux virtuels complémentaires sont disponibles. Un démon *ssh* est disponible pour les connexions à distance.

defclass : Appelle **fai-class**[(1)] pour définir les classes en utilisant les scripts et les fichiers dans `/fai/class` et des classes de `/tmp/fai/additional-classes` .

defvar : Source tous les fichiers `/fai/class/*`.var pour chaque classe définie. Si un *crochet* a écrit quelques définitions de variables dans le fichier `/tmp/fai/additional.var` , ce fichier est aussi sourcé.

action : Selon la valeur de `$FAI_ACTION` , ce sous-programme décide quelle action FAI doit exécuter. Les actions par défaut disponibles sont : sysinfo et install. Si `$FAI_ACTION` a une autre valeur, une action définie par l'utilisateur est appelée si un fichier `/fai/hooks/$FAI_ACTION` existe. Donc vous pouvez facilement définir vos propres actions.

sysinfo : Appelé quand aucune installation n'est exécutée, mais que l'action est sysinfo . Cela récupère les informations sur le matériel détecté et monte les disques durs locaux en lecture seule sur `/tmp/target/partitionname` ou en suivant le contenu d'un fichier `fstab` trouvé sur l'une des partitions. Les fichiers de log sont stockés sur le serveur de logs.

install : Cette tâche contrôle la séquence d'installation. Vous entendrez trois bips sonores avant que l'installation ne démarre. Le travail principal est d'appeler les autres tâches et de sauvegarder la sortie dans `/tmp/fai/rcS.log`. Si vous avez des problèmes pendant l'installation, regardez tous les fichiers contenus dans `/tmp/fai/`. Vous trouverez des exemples de fichiers de log pour quelques hôtes dans le répertoire de téléchargement de la page d'accueil FAI.

partition : Appelle **setup_harddisk** pour partitionner les disques durs. La tâche écrit les définitions de variables pour les partitions et device *root* et *boot* (`$ROOT_PARTITION` , `$BOOT_PARTITION` , `$BOOT_DEVICE`) dans `/tmp/fai/disk_var.sh` et crée un fichier `fstab`.

mountdisks : Monte les partitions, selon le fichier `/tmp/fai/fstab` créé, relativement à `$FAI_ROOT`.

extrbase : Extrait le fichier compressé `base.tgz` , qui contient tous les logiciels demandés. C'est une image d'un système *Debian* de base créé par **debootstrap**[(8)]

13 `/usr/share/fai/subroutines-linux` pour *Linux*, `/usr/share/fai/subroutines-sunos` for *Sloaris*.

mirror Si un miroir *Debian* local est accessible par NFS (quand `$FAI_DEBMIRROR` est définie), ce répertoire sera monté à `$MNTPOINT` .

updatebase Prépare le système de base *Debian* précédemment extrait pour l'installation et met à jour la liste de paquets disponibles. Met à jour la version des paquets. Elle modifie aussi quelques commandes (appelées *diversions*) à l'intérieur du nouveau système en utilisant **dpkg-divert**[(8)].

instsoft Installe les paquets souhaités en utilisant les fichiers de classes dans `/fai/package_config`.

configure Appelle les scripts de `/fai/scripts/` et ses sous-répertoires pour chaque classe définie.

finish Démonte tous les systèmes de fichiers dans le nouveau système et supprime les diversions de fichiers en utilisant la commande **fai-divert** .

faiend Attend que les taches de fond se terminent (par exemple la compilation des fichiers *Lisp* d'*Emacs*) et redémarre automatiquement le client ou attend une entrée manuelle avant de rebouter.

chboot Change le lien symbolique sur le serveur d'installation qui indique quelle image de noyau doit être chargée au prochain démarrage sur la carte réseau via TFTP.

savelog Sauvegarde les fichiers de log sur le disque local et sur les comptes `$LOGUSER` sur `$LOGSERVER` (par défaut sur le serveur d'installation). Actuellement le fichier `error.log` n'est pas copié sur le serveur de logs.

Les routines de paramétrage des clients

Après l'initialisation de base, réalisée par le sous-programme **fai_init** (création du disque virtuel, lecture de `fai.conf` et définitions de tous les sous-programmes, mise en place du « `path` », impression du *copyright*), l'installation continue en appelant la tâche **confdir** et la tâche **setup** . La commande **get-boot-info** est appelée pour obtenir toutes les informations du serveur BOOTP ou DHCP. Cette commande écrit le fichier `/tmp/fai/boot.log` , qui est alors sourcé pour définir les variables globales correspondantes. Voici un exemple de fichier de log avec utilisation d'un serveur DHCP.

```
# cat /tmp/fai/boot.log

netdevices_all="eth0
eth0 eth0"
netdevices_up="eth0"
netdevices="eth0"
BROADCAST='192.168.1.255'
DOMAIN='localdomain'
DNSSRVS='192.168.1.1'
DNSSRVS_1='192.168.1.1'
HOSTNAME='demohost'
IPADDR='192.168.1.12'
NETWORK='192.168.1.0'
GATEWAYS='192.168.1.250'
GATEWAYS_1='192.168.1.250'
SERVER='faiserver'
NETMASK='255.255.255.0'
```

On passe des informations complémentaires par la ligne de commande du noyau ou depuis le fichier `/etc/fai/fai.conf` . En démarrant avec PXE, les paramètres de ligne de commande sont créés en utilisant **fai-chboot**[(8)]. La variable `$FAI_FLAGS` contient une liste d'options séparées par un espace. On connaît les options suivantes :

verbose Fournit plus d'informations pendant le déroulement de l'installation. Cela devrait toujours être la première option, car ainsi les définitions des options suivantes sont affichées verbeusement.

debug Fournit des informations de mise au point. Aucune installation sans surveillance n'est exécutée. Pendant l'installation des logiciels, vous devez répondre à toutes les questions des scripts de post-installation sur la console du client. De nombreuses informations de debogage seropnt affichées. Cette option n'est utile que pour les développeurs FAI.

sshd Démarre le démon **ssh** pour permettre les connexions à distance.

syslogd Démarre les démons de logs système et noyau, ce qui permet aux processus de l'utiliser pour distribuer l'information. Cette option ne devrait être utilisée que si le **syslogd** n'est pas déjà actif sur le système, donc il ne peut être utilisé que sur l'installation initiale, pas sur un *update* !

createvt Créé deux terminaux virtuels et exécute un shell *bash* si **ctrl-c** est tapé sur la console. Les terminaux complémentaires peuvent être accédés en tapant **Alt-F2** ou **Alt-F3**. Sinon, aucun terminal n'est disponible et taper **ctrl-c** redémarrera le client. Mettre cette option est utile pour le débogage. Si vous voulez une installation qui ne soit pas interruptible, ne mettez pas ce drapeau.

reboot Redémarre le client lorsque l'installation est finie sans taper **Entrée** sur la console. Ce n'est utile que si vous pouvez changer d'image de boot ou d'unité de boot automatiquement ou si votre robot peut retirer la disquette de boot par contrôle à distance :-) Pour le moment, cela ne devrait être utilisé qu'avec le boot sur la carte réseau.

Le concept de classe

Les classes déterminent quel(s) fichier(s) de configuration choisir dans une liste de modèles disponibles. Les classes sont utilisées dans toutes les tâches de l'installation. Pour déterminer quelle configuration utiliser, un client recherche la liste des classes définies pour lui et utilise tous les fichiers de configuration qui correspondent à un nom de classe. Il est aussi possible de n'utiliser que le fichier de configuration avec la priorité la plus haute, puisque l'ordre de classes définit la priorité de la plus basse à la plus haute. Il y a quelques classes prédéterminées (DEFAULT, LAST et « *hostname* »), mais les classes peuvent aussi être inscrites dans un fichier ou définies dynamiquement par des scripts. Il est alors facile de définir une classe en fonction du sous-réseau ou d'un certain matériel présent sur la machine cliente.

L'idée d'utiliser les classes en général et d'utiliser certains fichiers correspondant à un nom de classe pour une configuration vient des scripts d'installation de Casper Dik pour *Solaris*. Cette technique s'est avérée très utile pour les postes de travail SUN, donc je l'utilise aussi pour l'installation entièrement automatique (FAI) de *Linux*. Une fonction simple et très efficace des scripts de Casper est d'appeler une commande avec tous les fichiers (ou seulement le premier) dont le nom soit aussi une classe.

La boucle suivante met en oeuvre cette fonction dans du code en pseudo-shell :

```
for class in $all_classes; do
if [ -r $config_dir/$class ]; then
   your_command $config_dir/$class
   # exit if only the first matching file is needed
fi
done
```

De cette façon, il est possible d'ajouter un nouveau fichier à la configuration sans changer le script. C'est parce que la boucle détecte automatiquement les nouveaux fichiers de configurations qui doivent être utilisés. Malheureusement *cfengine* ne supporte pas cette fonction agréable, alors toutes les classes utilisées dans *cfengine* doivent aussi être spécifiées à l'intérieur des scripts *cfengine*.

Les classes sont primordiales pour l'installation entièrement automatique.

Si un client appartient à la classe *A*, nous disons que la classe *A* est définie. Une classe n'a aucune valeur, elle est juste définie ou non définie. Dans des scripts, la variable `$classes` contient la liste des noms de toutes les classes définies, séparés par un espace.

Les classes déterminent comment l'installation est exécutée. Par exemple, un client peut être configuré pour devenir un serveur FTP en y ajoutant juste la classe FTP . Et surtout une configuration peut être créée en modifiant ou en ajoutant des classes auxquelles un client appartient, l'installation d'un nouveau client est alors très facile. Ainsi aucune information supplémentaire ne doit être ajoutée aux fichiers de configuration si les classes existantes suffisent pour vos besoins.

Il y a plusieurs possibilités différentes de définir des classes :

1. Quelques classes par défaut sont définies pour chaque hôte : DEFAULT, LAST et son «*hostname*».

2. Les classes peuvent être inscrites dans un fichier.

3. Les classes peuvent être définies par des scripts.

La dernière option est un dispositif très agréable, puisque ces scripts définiront des classes automatiquement. Par exemple, plusieurs classes sont définies seulement si un certain matériel est identifié. Nous utilisons *Perl* et des scripts *shell* pour définir des classes.

Tous les noms de classes, sauf pour «*hostname*» , sont écrits en majuscule. Ils ne doivent pas contenir de trait d'union, de dièse ou de point, mais peuvent contenir un souligné. Une description de toutes les classes peut être trouvée dans `/usr/share/doc/fai/classes_description.txt`. Le «*hostname*» devrait être rarement utilisé pour les fichiers de configuration dans l'espace de configuration. Il vaut mieux définir une classe et l'ajouter ensuite pour un hôte donné. En effet la plupart du temps les données de configuration ne sont pas spécifiques à un hôte, mais peuvent être partagées par plusieurs clients.

Définition de classes

La tâche par défaut **defclass** appelle le script **faiclass**[(1)] pour définir les classes. Les scripts correspondants à ***[0-9][0-9]**** dans `/fai/class` sont exécutés. De plus, les fichiers dans ce répertoire peuvent contenir une liste de classes. Nous utilisons par exemple un fichier `KOELN` pour tous nos clients qui appartiennent à un certain sous-réseau. Si nous voulons ajouter une classe à toutes ces machines, nous ajoutons juste la classe à ce fichier. Pour plus d'information sur la définition de classes, lisez les pages de manuel pour **faiclass**[(1)]. La liste de toutes les classes définies est stockée dans la variable `$classes` et sauvegardée dans `/tmp/fai/FAI_CLASSES` . La liste de toutes les classes est transférée à *cfengine* , donc il peut aussi les utiliser.

Le script **01alias** (ci-dessous en version dépouillée) est utilisé pour définir des classes pour plusieurs groupes de machines. D'abord ce script définit la classe avec le nom de l'architecture matérielle en lettres majuscules. Les hôtes qui ont une adresse IP dans le sous-réseau `134.95.9.0` appartiennent aussi à la classe `NET_9` , les hôtes dans le sous-réseau de classe B `134.95` utilisent toutes les classes du fichier `KOELN` . Tous les noeuds *Beowulf* avec le préfixe *atom* sauf `atom00` (le serveur maître) appartiendront aux classes inscrites dans le fichier `atoms` .

Voici un exemple de la simplicité avec laquelle vous pouvez grouper les machines qui doivent appartenir au même groupe de classes.

```
# cat 01alias

uname -s | tr /a-z/ /A-Z/
[ -x "`which dpkg`" ] && dpkg --print-installation-architecture | tr /a-z/ /A-Z/

# the Beowulf cluster; all nodes except the master node
# use classes from file class/atoms
case $HOSTNAME in
    atom00) echo BEOWULF_MASTER ;;
    atom??) cat atoms ;;
esac

# if host belongs to class C subnet 134.95.9.0 use class NET_9
# exclude all hosts with an IP address above 200
case $IPADDR in
    134.95.9.2??) ;;
    134.95.*.*) cat koeln ; echo "CS_KOELN NET_9" ;;
    134.95.9.*) echo "CS_KOELN NET_9" ;;
esac
```

Le script **24nis** définit automatiquement des classes correspondant au NIS. Le nom du domaine NIS (défini via BOOTP ou DHCP) devient aussi une classe (en lettres majuscules uniquement, et les tirets sont remplacés par un souligné). Si aucun domaine NIS n'est défini, seule la classe `NONIS` est définie.

En fonction des noms de partitions définis dans la première correspondance **disk_config** trouvée, **70partitions** définit des classes supplémentaires. Par exemple, si une partition `/files/scratch` existe, la classe `FILES_SCRATCH` est définie. Elle force le client à exporter ce répertoire par NFS et à installer les paquets du serveur NFS.

Le script **06hwdetect.source** utilise les commandes par défaut de *Debian* pour détecter le matériel SCSI et char-

ger les drivers nécessaires au noyau. Si un matériel spécifique est trouvé, il peut aussi définir une nouvelle classe pour celui-ci.

Vous pouvez trouver les messages de modprobe dans `/tmp/fai/kernel.log` et sur le terminal de la quatrième console en appuyant **Alt-F4**.

Définition de Variables

La tâche **defvar** définit les variables pour le client. Les variables sont définies par des scripts dans `class/*.var`. Toutes les variables globales peuvent être déclarées dans **DEFAULT.var** . Pour certains groupes de machines, il faut utiliser un fichier de classe, mais pour un hôte isolé, on utilise le fichier ***hostname*.var** . Là encore, il est utile d'étudier tous les exemples. Les variables suivantes sont utilisées dans les exemples et peuvent être très utiles pour personnaliser votre installation :

FAI_ACTION — indique l'action que FAI doit exécuter. Normalement cela est fait par **fai-chboot**[(8)]. Si vous ne pouvez pas utiliser cette commande et que vous n'utilisez pas de serveur BOOTP, définissez cette variable dans le script **LAST.var**.

FAI_CONSOLEFONT — désigne la police qui est chargée pendant l'installation par **consolechars**[(8)].

FAI_KEYMAP — définit les fichiers de map du clavier dans `/usr/share/keymaps` et *`$FAI`*`/files` . Vous n'avez pas besoin de spécifier le chemin complet, ce fichier sera placé automatiquement.

rootpw — Le mot de passe de «root» pour le nouveau système. De plus, FAI crée un compte «root» avec le même mot de passe appelé «rootf» , qui utilise tcsh[(1)].

UTC — règle l'horloge matérielle sur UTC si `$UTC=yes` . Sinon, l'horloge est réglée sur l'heure locale. Voir **clock**[(8)] pour plus d'information.

time_zone — est le fichier relatif à `/usr/share/zoneinfo/` qui indique votre fuseau horaire.

liloappend — ajoute des paramètres pour le noyau du nouveau système (écrit dans `/etc/lilo.conf`).

moduleslist — peut être une définition multi-lignes. C'est la liste des modules (en incluant les paramètres de noyau) qui sont chargés pendant le boot du nouveau système (écrit dans `/etc/modules`).

TFTPLINK — assure la liaison vers l'image de noyau TFTP au démarrage en utilisant le système de fichiers racine du disque local. C'est utilisé uniquement avec un serveur BOOTP pour le démarrage.

hserver, bserver — les noms des serveurs NFS pour `/home` et `/usr` ou `/usr/local`.

printers — la liste des imprimantes pour lesquelles un répertoire de spool est créé. Les scripts de configuration n'initialisent pas `/etc/printcap`.

addpackages — contient la liste des paquets complémentaires qui sont installés sur le nouveau système, s'ils sont disponibles dans `/fai/files/packages`. Elle doit aussi contenir le nom du paquet du noyau qui doit être installé. Vous pouvez créer un dépôt simple en utilisant les commandes suivantes sur le serveur d'installation :

```
# cd /usr/local/share/fai/files
# dpkg-scanpackages packages /dev/null | \
  gzip -9 > packages/Packages.gz
```

Si vous ne voulez pas utiliser cette fonction (pour quelle raison ?), créez un fichier vide `Packages` pour supprimer les messages d'erreurs. En complément, vous pouvez aussi créer un fichier de version (*Release*) dans ce répertoire. Alors `addpackages` peut être la liste des paquets sans numéro de version. Pour plus d'information, référez-vous au «repository-howto»[14]. Ceci peut être utilisé pour installer des paquets spécifiques au site.

[14] http://www.isotton.com/debian/docs/repository-howto/

Configuration du disque dur

Le format des fichiers de configuration de disque dur est décrit dans `/usr/share/doc/fai/README.disk_config.gz`.

Le fichier de configuration `/fai/disk_config/CS_KOELN` est une description générique pour un disque dur IDE, qui devrait convenir pour la plupart des installations. Si vous ne pouvez pas partitionner votre disque dur en utilisant ce script[15], utilisez un «crochet» à la place. Le «crochet» doit écrire la nouvelle table de partition, créer les systèmes de fichiers et écrire les fichiers `/tmp/fai/fstab` et `/tmp/fai/disk_var.sh` , qui contiennent les définitions des partitions racine et de boot.[15]

Configuration des paquets logiciels

Le script **install_packages** installe les logiciels sélectionnés. Il utilise tous les fichiers de configuration présents dans `/fai/package_config` dont le nom correspond à une classe définie. La syntaxe est très simple.

```
# an example package class

PACKAGES taskinst
german science

PACKAGES install
adduser netstd ae
less passwd

PACKAGES remove
gpm xdm

PACKAGES dselect-upgrade
ddd                     install
a2ps                    install
```

Les commentaires commencent par un dièse (*#*) et se terminent à la fin de la ligne. Chaque commande commence par le mot *PACKAGES* suivi par un nom de commande. Le nom de commande est similaire à ceux **apt-**d'**get** . Voici la liste de noms de commande acceptés :

hold	Fige l'état d'un paquet. Ce paquet ne sera pas traité par **dpkg**, par exemple ne sera pas upgradé.
install	Installe tous les paquets qui sont spécifiés dans les lignes suivantes. Si un trait d'union (-) est ajouté devant le nom du paquet (sans espace intercalaire), le paquet sera retiré, et non pas installé. L'orthographe de tous les noms de paquet est vérifiée. Tous les paquets qui n'existent pas, seront enlevés de la liste de paquets à installer. Alors faites attention pour ne pas faire d'erreur sur le nom d'un paquet.
remove	Enlève tous les paquets qui sont spécifiés dans les lignes suivantes. Ajoutez un (+) au nom du paquet si le paquet doit être installé.
taskinst	Installe tous les paquets appartenant aux tâches qui sont spécifiées dans les lignes suivantes, en utilisant **`tasksel`**[(1)].
dselect-upgrade	positionne les sélections de paquets en fonction des lignes suivantes et installe ou retire les paquets indiqués. Ces lignes sont produites par la commande **dpkg –get-selections** .

De nombreuses lignes avec les listes de noms de paquets séparés par des espaces s'inscrivent à la suite des commandes **install** et **remove** . Toutes les dépendances sont résolues et **apt-get** est utilisé pour exécuter l'installation ou la suppression des paquets. L'ordre des paquets importe peu.

Une ligne qui contient la commande **PRELOADRM** télécharge un fichier en utilisant **wget**[(1)] dans un répertoire avant d'installer les paquets. En utilisant l'URL *file:* , ce fichier est copié depuis `$FAI_ROOT` vers le répertoire de téléchargement. Par exemple le paquet `realplayer` a besoin d'une archive pour installer le logiciel, alors cette archive est téléchargée dans le répertoire `/root` . Après l'installation des paquets ce fichier sera supprimé. Si le fichier ne doit pas être enlevé, il faut utiliser la commande **PRELOAD** à la place .

[15]Actuellement le script utilise la commande **sfdisk**[(8)], qui n'est pas disponible sur SUN™ SPARC™ et IA64.

Maintenant il est possible d'ajouter une liste de noms de classes après la commande pour **apt-get** . Alors la commande **PACKAGE** ne sera exécutée que si la classe correspondante est définie. Ainsi vous pouvez combiner beaucoup de petits fichiers dans le fichier `DEFAULT`. ATTENTION ! N'utilisez cette fonction que dans le fichier `DEFAULT` pour conserver les choses simples. Reportez-vous à ce fichier pour quelques exemples.

Si vous indiquez un paquet qui n'existe pas, ce paquet sera enlevé de la liste d'installation. Vous pouvez aussi tester tous les fichiers de configuration de logiciels avec l'utilitaire **chkdebnames** , qui est disponible dans `/usr/share/doc/fai/examples/utils/`.

```
> chkdebnames stable /usr/local/share/fai/package_config/*
```

Scripts dans /fai/scripts

L'ensemble de scripts par défaut dans `/fai/scripts` est fournit uniquement à titre d'exemple. Mais ils devraient raisonnablement convenir pour votre installation. Vous pouvez les éditer ou ajouter de nouveaux scripts pour correspondre à vos besoins.

La commande **fai-do-scripts**[(1)] est appelée pour exécuter tous les scripts dans ce répertoire. Si un répertoire portant un nom de classe existe, tous les scripts du type `S[0-9]*` sont exécutés par ordre alphabétique. Il est ainsi possible d'utiliser des scripts de langages différents (*shell, cfengine, perl*...) pour chaque classe.

Scripts Shell

La plupart des scripts sont des scripts BASH . Les script *Shell* sont très utiles si la tâche de configuration doit seulement appeler quelques commandes de shell ou créer un fichier à partir de zéro. De façon à ne pas écrire plein de petits scripts, il est possible de distinguer les classes à l'intérieur d'un script en utilisant la commande **ifclass** . Pour copier des fichiers en rapport avec des classes, utilisez la commande **fcopy**[(8)]. Si vous voulez extraire des archives en utilisant des classes, utilisez **ftar**[(8)]. Mais maintenant regardez les script pour voir ce qu'ils font.

Scripts Perl

Actuellement aucun script *Perl* n'est utilisé pour modifier la configuration de système.

Expect scripts

Actuellement aucun script «*Expect*» n'est utilisé pour modifier la configuration de système.

Scripts Cfengine

Cfengine possède un jeu de fonctions assez riche pour éditer des fichiers de configuration existants, par exemple `LocateLineMatching` , `ReplaceAll` , `InsertLine` , `AppendIfNoSuchLine` , `HashCommentLinesContaining` . Mais il ne peut pas manipuler de variables qui ne sont pas définies. Si une seule variable n'est pas définie, la totalité du script *cfengine* échouera. Étudiez les exemples fournis dans le paquet *fai* . Vous trouverez plus d'information dans la page de manuel *cfengine*[(8)] ou sur la page d'accueil *cfengine* http://www.cfengine.org [http://www.cfengine.org/].

Changer l’unité de démarrage

Le changement de l'ordre de boot est normalement réalisé dans le setup du BIOS. Mais on ne peut pas changer le BIOS depuis une session Linux active (pour autant que je le sache !) Si vous savez le faire, envoyez-moi s'il vous plaît un e-mail. Mais il y a une autre façon d'échanger le dispositif de boot d'un système *Linux* en fonctionnement.

Normalement, l'ordre de boot du BIOS reste inchangé et votre ordinateur doit toujours démarrer en premier sur sa carte réseau et en second sur le disque dur local. On peut alors obtenir une image de noyau d'installation à partir du serveur d'installation, si on exécute une installation, ou on peut dire à **pxelinux** de démarrer sur le disque local. C'est ce qu'on fait en utilisant **fai-chboot**[(8)].

Voici comment configurer une carte réseau 3Com™ comme premier dispositif de boot. Validez le réseau local

comme premier dispositif de boot dans le BIOS.

```
Boot From LAN First: Enabled Boot Sequence : C only
```

Entrez alors dans le setup MBA de la carte réseau 3COM™ et changez-le comme suit :

```
Default Boot Local Local Boot Enabled Message Timeout 3 Seconds Boot Fai-
lure Prompt Wait for timeout Boot Failure Next boot device
```

Cela validera le premier disque dur IDE comme second dispositif de boot. En boutant sur une disquette FAI, vous avez une autre solution pour éviter une réinstallation si le BIOS est configuré pour démarrer sur la disquette en premier et que vous n'êtes pas là pour enlever la disquette :

```
# lilo -R ...
```

indiquera à la disquette FAI de démarrer du disque dur une seule fois (voir **lilo** [(8)]). Ainsi après ce premier reboot, la disquette FAI peut être utilisée pour une autre installation FAI.

Crochets (Hooks)

Les «crochets» vous permettent de spécifier des fonctions ou les programmes qui sont exécutés à certaines phases du processus d'installation. Avant d'appeler une tâche par défaut, FAI recherche l'existence de «crochets» pour cette tâche et les exécute. Comme vous vous en doutez, les classes sont aussi utilisées pour appeler les «crochets». Les «crochets» sont exécutés pour chaque classe définie. Vous devez seulement créer le «crochet» avec le nom de la classe désirée et il sera utilisé. Si l'option `debug` est incluse dans `$FAI_FLAG`, on passe l'option `-d` à tous les crochets, vous pouvez donc mettre au point vos propres «crochets». Si vous devez sauter certaines tâches par défaut, utilisez le sous-programme **skiptask** avec la liste de ces tâches par défaut comme paramètres. L'exemple **partition.DISKLESS** saute certaines tâches par défaut.

Le répertoire `/fai/hooks/` contient tous les «crochets». Le nom de fichier d'un «crochet» consiste en un nom de «crochet» comme préfixe et un nom de classe, séparés par un point. Le préfixe décrit le moment où le «crochet» est appelé, si la classe est définie pour le client. Par exemple, le «crochet» **partition.DISKLESS** est appelé pour chaque client appartenant à la classe `DISKLESS` avant le partitionnement des disques locaux. Si cela doit devenir un client sans disque, ce «crochet» peut monter les systèmes de fichiers distants via NFS et créer `/tmp/fai/fstab` . Ensuite, le processus d'installation n'essayera pas de partitionner et formater un disque dur local, parce qu'un fichier `/tmp/fai/fstab` existe déjà.

Un «crochet» de la forme ***`hookprefix.classname`*** ne peut pas définir de variables pour le script d'installation, parce que c'est un sous-programme. Mais vous pouvez utiliser n'importe quel fichier binaire exécutable ou n'importe quel script que vous avez écrit. Les «crochets» qui ont le suffixe *.source* (par exemple **partition.DEFAULT.source**) doivent être des script BASH et sont sourcés. De cette façon, il est possible de redéfinir des variables pour les script d'installation.

Dans la première partie de *fai*, tous les «crochets» avec le préfixe *confdir* sont appelés. Comme le répertoire de configuration `/fai` est monté dans la tâche par défaut **confdir** , les «crochets» pour cette tâche sont les seuls «crochets» placés dans `$nfsroot/fai/hooks` sur le serveur d'installation. Tous les autres «crochets» se trouvent dans `/usr/local/share/fai/hooks` sur le serveur d'installation. Tous les «crochets» qui sont appelés avant que les classes ne soient définies ne peuvent utiliser que les classes suivantes : `DEFAULT` ; *`$HOSTNAME`* ; `LAST` . Si un crochet pour la classe `DEFAULT` ne doit être appelé que si aucun crochet pour la classe `$HOSTNAME` n'est disponible, insérez ces lignes dans le crochet par défaut :

```
hookexample.DEFAULT:

#! /bin/sh

# skip DEFAULT hook if a hook for $HOSTNAME exists
scriptname=$(basename $0 .DEFAULT)
[-f /fai/hooks/$scriptname.$HOSTNAME ] && exit
# here follows the actions for class DEFAULT
.
.
```

Quelques exemples d'utilisations possibles des crochets :

- Utiliser **ssh** au tout début pour vérifier que vous avez monté la configuration sur le bon serveur et pas sur un possible hôte de spoofing.
- Ne pas monter le répertoire de configuration, mais au lieu de cela récupérer des archives compressées via HTTP ou sur une disquette et l'extraire dans un nouveau *ram-disque*, puis redéfinir `$FAI_LOCATION`.
- Charger des modules de noyau avant que les classes ne soient définies dans `/fai/class`.
- Envoyer un e-mail à l'administrateur lorsque l'installation est terminée.
- Installer un client sans disque et éviter le partitionnement de disque local. Voir `hooks/partition.DISKLESS`.
- Partitionner le disque dur sur un système IA64, qui a besoin d'un type spécial de table de partition qui doit être créé avec **parted**[(8)]. Voir `hooks/partition.IA64`.

Recherche les erreurs

Si le client ne peut pas bouter sur la carte réseau, utilisez **tcpdump** pour regarder les paquets *Ethernet* entre le serveur d'installation et le client. Recherchez aussi les entrées dans plusieurs fichiers de log créées par **in.tftpd**[(8)], **dhcpd3**[(8)]ou **bootpd**[(8)]:

```
egrep "tftpd|bootpd|dhcpd" /var/log/*
```

Si le processus d'installation se termine, le «crochet» **faiend.LAST** recherche les erreurs communes dans tous les fichiers de log et les écrit dans le fichier `error.log`. Vous devrez donc d'abord examiner ce fichier pour voir les erreurs. Le fichier `status.log` vous donne le code de sortie de la dernière commande exécutée dans un script. Pour être sur, vous devriez chercher les erreurs dans tous les fichiers de log.

Parfois l'installation semble s'arrêter, mais c'est seulement un script de *postinstall* d'un logiciel qui demande une entrée manuelle sur la console. Passez sur un autre terminal virtuel et regardez quel processus est actif avec des outils comme **top**[(1)]et **pstree**[(1)] . Vous pouvez ajouter `debug` à `FAI_FLAGS` pour que le processus d'installation montre toute les sorties des scripts de *postinstall* sur la console et établir aussi son entrée sur la console. N'hésitez pas à envoyer un e-mail à la liste de diffusion ou à `<fai@informatik.uni-koeln.de>` si vous avez des questions. Des fichiers de log exemples d'ordinateurs installés avec succès sont disponibles sur la page d'accueil FAI.

Comment construire un cluster Beowulf en utilisant FAI

Ce chapitre décrit en détails la construction d'un cluster *Beowulf* en utilisant *Debian GNU/Linux* et FAI. Pour plus d'information sur le concept *Beowulf*, consulter : http://www.beowulf.org [http://www.beowulf.org/].

Préparation de la configuration Beowulf

Le cluster *Beowulf* de l'exemple est constitué d'un noeud maître et de 25 clients. Tout le matériel est assemblé dans un grand rack, avec un clavier et un moniteur. Ils sont connectés au serveur maître la plupart du temps, mais comme les câbles du moniteur et du clavier sont longs, ils peuvent ainsi être connectés sur chacun des noeuds si une modification doit être faite dans le BIOS, ou pour le dépannage quand un noeud ne démarre pas.

L'alimentation en énergie doit aussi faire l'objet d'une étude. Il ne faut pas brancher trop de noeuds sur chaque cordon d'alimentation et sur chaque prise. Répartissez-les sur plusieurs boîtes de dérivations et prises.

Et en ce qui concerne l'émission de chaleur ? Une douzaine de noeuds dans une petite pièce peuvent créer trop de chaleur, il faut alors prévoir un climatiseur.

Après une coupure d'alimentation électrique, les alimentations de chaque noeuds basculent-elles en *stand-by*, ou tous les noeuds redémarrent-ils simultanément ?

Tous les ordinateurs dans cet exemple sont connectés à un *switch Fast Ethernet*. Le noeud principal (ou serveur maître) est appelé «nucleus». Il est équipé de deux cartes réseau. Une pour la connexion à l'Intranet externe, une pour la connexion au réseau interne du cluster. Pour une connexion depuis l'Intranet externe, il s'appelle «nucleus», mais les noeuds de cluster accèdent au noeud maître par le nom «atom00». C'est le nom de la deuxième interface réseau.

Le serveur maître est aussi le serveur d'installation pour les noeuds de calcul. Un miroir *Debian* local sera installé sur son disque dur. Le répertoire `/home` de chaque compte utilisateur est aussi placé sur le serveur maître. Il sera exporté via NFS à tous les noeuds de calcul. NIS sera utilisé pour distribuer le compte et les informations de noms machines et d'imprimantes à tous les noeuds.

Tous les noeuds clients *atom01* à *atom25* sont connectés via le *switch* sur la deuxième carte d'interface du noeud maître. Ils ne peuvent se connecter qu'aux autres noeuds ou au maître, mais ne peuvent communiquer avec aucun hôte à l'extérieur du réseau du cluster. Ainsi, tous les services (NTP, DNS, NIS, NFS...) doivent être disponibles sur le serveur maître. Je choisis le réseau de classe 'C' *`192.168.42.0`* pour adresser le réseau du cluster *Beowulf* local. Vous pouvez remplacer le sous-réseau *`42`* par un autre nombre si vous préférez. Si vous avez plus de 253 noeuds de calcul, choisissez un réseau de classe 'A' *`10. X.X.X`*.

Dans la phase de préparation de l'installation, vous devrez démarrer plusieurs fois le premier client, jusqu'à ce qu'il n'y ait aucune erreur dans vos script de configuration. Vous devrez donc avoir un accès physique au serveur maître et au noeud client. Alors, connectez les deux ordinateurs à un commutateur, le clavier et le moniteur sont ainsi partagés par les deux machines.

Configurer le serveur maître

Le serveur maître sera installé à la main si c'est votre premier ordinateur installé avec *Debian*. Si vous avez déjà un hôte sous *Debian*, vous pouvez aussi installer le serveur maître via FAI. Créez une partition sur `/files/scratch/debmirror` pour le miroir *Debian* local avec plus de 9.0 GB d'espace disponible.

Configurer le réseau

Ajoutez les lignes suivantes dans `/etc/network/interfaces` pour la seconde carte réseau :

```
# Beowulf cluster connection
auto eth1
iface eth1 inet static
address 192.168.42.250
netmask 255.255.255.0
broadcast 192.168.42.255
```

Ajoutez les adresses IP pour les noeuds clients. Le paquet FAI propose un exemple pour le fichier `/etc/hosts` :

```
# create these entries with the perl one liner
# perl -e 'for (1..25) {printf "192.168.42.%s atom%02s\n",$_,$_;}'

# Beowulf nodes
# atom00 is the master server
192.168.42.250 atom00
192.168.42.1 atom01
192.168.42.2 atom02
```

Vous pouvez donner un nom au réseau interne du cluster en ajoutant cette ligne au fichier `/etc/networks` :

```
beowcluster 192.168.42.0
```

Activer la deuxième interface réseau :

```
# /etc/init.d/networking start
```

Configuration NIS

Ajoutez un utilisateur normal nommé par exemple *Tom*, qui est la personne qui édite l'espace de configuration et

gère le miroir *Debian* local :

```
# adduser tom
# addgroup linuxadmin
```

Cet utilisateur doit aussi être dans le groupe *linuxadmin* .

```
# adduser tom linuxadmin
```

Choisissez d'abord le nom de domaine NIS en créant le fichier `/etc/defaultdomain` , puis en appelant **domainname**[(8)]. Initialisez le serveur maître comme serveur NIS en appelant :

```
# /usr/lib/yp/ypinit -m
```

Éditez aussi `/etc/default/nis` pour que l'hôte devienne un serveur de domaine NIS. Copiez alors le fichier `netgroup` du répertoire contenant les exemples dans `/etc` et éditez le. Paramétrez l'accès au service NIS.

```
# cat /etc/ypserv.securenets
# Always allow access for localhost
255.0.0.0       127.0.0.0
# This line gives access to the Beowulf cluster
255.255.255.0 192.168.42.0
```

Reconstruisez les maps NIS :

```
# cd /var/yp; make
```

Vous trouverez beaucoup plus d'information sur NIS dans le document *NIS-HOWTO*.

Créer un miroir local Debian

Maintenant l'utilisateur *Tom* peut créer un miroir *Debian* local sur `/files/scratch/debmirror` en utilisant **mkdebmirror**. Vous pouvez ajouter l'option *`-debug`* pour voir quels fichiers sont reçus. Il y aura besoin de 9.0 GB d'espace disque pour *Debian 3.0* (c'est-à-dire *woody*). Exportez ce répertoire au netgroup *@faiclients* en lecture seule. Voici un exemple pour le fichier `/etc/exports`

```
/files/scratch/debmirror * (ro)
```

Installer le paquet FAI sur le serveur maître

Ajoutez les paquets suivants au serveur d'installation :

```
nucleus:/# apt-get install ntp tftpd-hpa dhcp3-server \
nfs-kernel-server etherwake fai fai-kernels
nucleus:/# tasksel -q -n install dns-server
nucleus:/# apt-get dselect-upgrade
```

Configurez NTP pour que le serveur maître ait le temps système correct. Il est très important d'utiliser le nom réseau *atom00* interne pour le serveur maître (pas le nom externe *nucleus*) dans `/etc/dhcp3/dhcpd.conf` et `/etc/fai/make-fai-nfsroot.conf`. Remplacez les termes *`FAISERVER`* par *atom00* et décommentez la ligne suivante dans `/etc/fai/make-fai-nfsroot.conf` pour que les noeuds *Beowulf* puissent utiliser ce nom pour joindre leur serveur maître.

```
NFSROOT_ETC_HOSTS="192.168.42.250 atom00"
```

Préparer le démarrage sur le réseau

Paramétrez le démon du serveur d'installation comme décrit dans Démarrage avec une carte réseau respectant la norme PXE.

Si vous devez connecter beaucoup de noeuds de cluster (plus qu'environ 10) et si vous avez choisi d'utiliser **rsh**

dans `/etc/fai/fai.conf`, augmentez le nombre de connections par minute à certains services dans `inetd.conf` :

```
shell stream tcp  nowait.300  root /usr/sbin/tcpd /usr/sbin/in.rshd
login stream tcp  nowait.300  root /usr/sbin/tcpd /usr/sbin/in.rlogind
```

L'utilisateur *Tom* doit avoir les droits pour créer les liens symboliques pour démarrer via la carte réseau, il faut alors changer le groupe et ajouter quelques utilitaires.

```
# chgrp -R linuxadmin /boot/fai; chmod -R g+rwx /boot/fai
# cp /usr/share/doc/fai/examples/utils/* /usr/local/bin
```

Maintenant, l'utilisateur Tom sélectionne l'image de *boot* pour le premier noeud *Beowulf*.

```
fai-chboot -IFv atom01
```

Démarrez maintenant le premier noeud client pour la première fois. Commencez alors à ajuster la configuration pour vos noeuds. N'oubliez pas de construire le noyau pour les noeuds de cluster en utilisant **make-kpkg**[(8)] et de l'enregistrer dans `/usr/local/share/fai/files/packages`.

Outillage pour les clusters Beowulf

Les outils suivants sont utiles pour un cluster Beowulf :

tlink — Change le lien symbolique qui pointe sur l'image du noyau utilisé pour démarrer à partir d'une carte réseau. Ce n'est utilisé que quand vous démarrez en utilisant BOOTP.

all_hosts — Imprime une liste de tous les hôtes, imprime seulement les hôtes qui répondent à un **ping** ou les hôtes qui ne répondent pas. La liste complète des hôtes est définie par le *netgroup* `allhosts`. Regardez un exemple dans `/usr/share/doc/fai/examples/etc/netgroup`.

rshall — Exécute via *rsh* une commande sur tous les hôtes qui sont actifs. Utilise **all_hosts** pour avoir la liste de tous les hôtes actifs. Vous pouvez aussi utiliser la commande **dsh**[(1)] (*dancer's shell*, ou *distributed shell*).

Voici ensuite quelques outils communs pour un environnement de *cluster* :

rgang — Pour un immense cluster, essayez **rgang**. C'est un outil qui exécute des commandes ou distribue des fichiers sur de nombreux noeuds. Il utilise un algorithme pour construire une structure en arbre qui permet la répartition du temps de traitement à l'échelle de 1000 noeuds ou plus (téléchargeable à http://fermitools.fnal.gov/abstracts/rgang/abstract.html).

jmon — Pour surveiller les ressources de tous les noeuds (CPU, mémoire, swap...) vous pouvez utiliser **jmon**[(1)] qui installe un simple démon sur chaque noeud de cluster.

ganglia — Ce *toolkit* est excellent pour contrôler votre cluster. Disponible à http://ganglia.sourceforge.net/.

Mais il y a beaucoup d'autres outils disponibles qui n'ont pas encore été inclus dans un paquet *Debian*.

«Wake on LAN» avec les cartes réseau 3Com™

«Wake on LAN» est une fonction très agréable sur un réseau local pour démarrer un ordinateur sans y avoir un accès physique. En envoyant un paquet éthernet spécial à la carte réseau, l'ordinateur s'allumera. Il faut réaliser les opérations suivantes pour utiliser la fonction «Wake on LAN» (WOL).

1. Connecter la carte réseau au connecteur «Wake on LAN» sur la carte mère utilisant le câble 3 broches.

2. Ma carte mère ASUS™ K7M a un cavalier appelé «Vaux» (*3VSBSLT*) qui permet de choisir la tension

fournie pour les cartes PCI additionnelles. Positionnez-le sur *3VSB* (*3 volts stand-by*).

3. Activez la fonction «Wake on LAN» dans le BIOS

4. Pour une carte 3Com™ utilisant le driver *3c59x*, vous devez valider la fonction de WOL en utilisant l'option de module du noyau `enable_wol`. Pour réveiller un ordinateur, utilisez la commande **ether-wake**[(8)]. Vous trouverez des informations complémentaires sur http://www.scyld.com/expert/wake-on-lan.html.

FAI sur d'autres architectures

Voici les choses qui doivent être modifiées sur les autres architectures :

task partition : Normalement, `setup_harddisks` requiert la commande **sfdisk** [(8)]. Si elle n'existe pas, ecrivez un court script shell en utilisant **parted** [(8)], pour partitionner le disque et pour créer le fichier `fstab`.

Boot loader Le script `DEFAUTLT/S10` supporte **lilo**[(8)] et **grub** [(8)]. Vous devrez ajouter le support de votre chargeur de boot spécifique.

FAI sur AMD64

Pas de problème, voir : http://www.informatik.uni-koeln.de/fai/download/amd64/

FAI sur PowerPC

FAI sur IA64

Il existe un gros cluster *Beowulf* IA64 qui fonctionne après avoir été installé avec FAI. Il n'y a que la fonction de partitionnement qui ait été remplacée par un court script, puisque sfdisk n'est pas disponible sur IA64.

FAI pour Suse, Redhat et Gentoo

De nombreuses personnes sont intéresées par FAI pour d'autres distributions Linux (basées principalement sur rpm). J'ai fait quelques recherches et celà ne devrait pas être un trop gros travail pour le réaliser. Mais j'ai besoin de plus d'aide pour l'implémenter. Si vous êtes intéressé et souhaitez m'aider, envoyez moi un mail à `<fai@informatik.uni-koeln.de>`

FAI sur du matériel SUN™ SPARC™ sous Linux

Bien que FAI soit indépendant de l'architecture, il y a quelques paquets qui ne sont disponibles que pour certaines architectures (par exemple *silo*, *sparc-utils*). Les ordinateurs SUN™ SPARC™ peuvent démarrer depuis la console de *boot* et n'ont pas besoin d'une disquette de *boot*. Pour démarrer, un SUN™ utilise :

```
boot net:dhcp - ip=::::::dhcp
```

Vous devez convertir l'image du noyau du format *ELF* au format *a.out*. Utilisez le programme **elftoaout** (mentionné dans la FAQ). Le lien symbolique vers l'image de noyau de boot n'est pas le nom d'hôte. Regardez la FAQ sur http://www.ultralinux.org et http://www.sparc-boot.org/ pour plus d'information. Un rapport de réalisation est disponible sur http://www.opossum.ch/fai/ et un HOWTO avec plein d'exemples peut être trouvé sur http://toolbox.rutgers.edu/~amurphy/fai.

FAI pour Solaris™

FAI est aussi porté pour être utilisé pour les installations de l'OS SUN™ Solaris™. Il est utilisé en coopération avec Solaris™ **jumpstart**. Chargez les sources de FAI et placez-vous dans le répertoire à `sunos`. Là vous pouvez appeler **make** qui crée le *tarball* `/tmp/fai-solaris.tar.gz`. Vous devrez lire le fichier `README.sunos` et avoir un peu de connaissance sur Solaris™ **jumpstart**. Le format des fichiers de configuration dans `/disk_config` et `/package_config` est différent que ceux pour *Linux*.

Utilisation de CVS avec FAI

Utiliser le contrôle de révision pour la configuration de FAI

Si c'est une équipe d'administrateurs qui est impliquée, un système de management de code source et de contrôle de révision comme CVS peut simplifier la coordination : plusieurs personnes peuvent travailler simultanément sur les fichiers de configuration, tandis que ce système aide à éviter les conflits (et s'il s'en produit, il aide à les résoudre). Un autre avantage se trouve dans la gestion des branches : pendant que l'administrateur prépare une nouvelle configuration et réalise les essais en utilisant une configuration de test, les autres clients ne subissent aucune perturbation, car ils utilisent une autre branche de la configuration.

Paramétrer FAI pour une configuration basée sur CVS

Vous devez d'abord initialiser un dépôt CVS et dans celui-ci, un module pour stocker les fichiers de configuration de FAI. Dans cet exemple, un serveur CVS «pserver» sera utilisé pour les accès en lecture seule aux fichiers de configuration par les clients, tandis que **ssh** est utilisé pour les accès des développeurs
(rw)[16]
.

Les variables correspondantes à CVS dans `/etc/fai/fai.conf` et `/etc/fai/make-fai-nfsroot.conf` sont :

FAI_LOCATION Cette variable ne doit pas être positionnée si vous voulez utiliser CVS.

FAI_CVSROOT contient le «cvsroot» où est stockée la configuration.

```
FAI_CVSROOT=":pserver:client@cvs.local.net:/var/lib/cvs"
```

FAI_CVSMODULE contient le module où est stockée la configuration dans le «cvsroot».

```
FAI_CVSMODULE="config"
```

FAI_CVSTAG contient le «tag» de la branche CVS qui sera extraite par le client[17].

```
FAI_CVSTAG="STABLE"
```

Si vous utilisez un serveur «pserver» pour stocker les fichiers de configuration, le fichier `/root/.cvspass` doit exister et être valide dans la «nfsroot». CVS utilise ce fichier pour obtenir le mot de passe pour le «pserver». Vous pouvez le créer très facilement si vous exécutez

```
cvs -d$FAI_CVSROOT login
```

et que vous copiez ensuite la ligne générée de votre `~/.cvspass` dans `/root/.cvspass` sur la «nfsroot».

16 17

16 *CVS* est très flexible et peut être utilisé avec plusieurs méthodes d'accès, c'est pourquoi je vous recommande de lire plus de documentation sur le sujet pour trouver la solution optimale dans votre environnement.
17 Ceci n'est pas obligatoire : si la variable n'est pas définie, *HEAD* sera utilisé, ce qui correspond à la version la plus récente de la configuration.

Points divers

Ce chapitre regroupe des points divers qui ne sont pas toujours expliqués dans le détail.

En utilisant l'accès HTTP à un miroir *Debian*, la partition locale `/var` sur tous les clients doit être assez grande pour contenir tous les paquets Debian téléchargés. N'essayez pas avec moins de 250 Mo à moins d'avoir une bonne raison. Vous pouvez limiter le nombre de paquets installés à la fois avec la variable `$MAXPACKAGES` .

Vous pouvez raccourcir certains scripts si vous allez juste utiliser une simple commande **fcopy**.

```
fcopy -r /.
```

Vous pouvez fusionner deux répertoires qui contiennent des informations de configuration, si l'un est global et l'autre local. Nous l'utilisons pour fusionner les modèles du paquet fai et notre configuration locale, qui contient les mots de passe chiffrés et d'autres informations qui ne doivent pas être lisible par d'autres. C'est à cela que notre installation ressemble.

Nous avons un espace de configuration local placé dans `~admin/additional-fai/` qui contient les fichiers suivants :

```
./files
./files/etc
./files/etc/hosts
./files/etc/hosts/NUERBURG1
./files/etc/hosts/NUERBURG2
./files/etc/network
./files/etc/network/interfaces
./files/etc/network/interfaces/NUERBURG1
./files/etc/network/interfaces/NUERBURG2
./files/etc/bootptab
./files/etc/bootptab/kueppers
./files/etc/kueppers.tar.gz
./files/packages
./files/packages/kernel-image-2.4.20-cskoeln_2_i386.deb
./files/packages/cloop-2.4.20-cskoeln_0.63.1-4+2_i386.deb
./files/packages/xv-doc_3.10a-26_all.deb
./files/packages/xv_3.10a-26_i386.deb
./files/packages/Packages.gz
./files/packages/ltmodem-2.4.20_8.26a9_i386.deb
./files/packages/cloop-2.4.20-acer_0.63.1-4+1_i386.deb
./files/packages/kernel-image-2.4.20-acer_1_i386.deb
./files/packages/pcmcia-modules-2.4.20-acer_3.1.33-6+1_i386.deb
./files/packages/kernel-headers-2.4.20-acer_1_i386.deb
./files/packages/ltmodem-2.4.20-acer_8.26a9_i386.deb
./files/packages/debmirror_20020427-1_all.deb
./files/usr
./files/usr/local
./files/usr/local/ACROREAD5.tar.gz
./files/usr/local/share
./files/usr/local/share/LPRng
./files/usr/local/share/LPRng/pcfilter
./files/usr/local/share/LPRng/pcfilter/NISLPRCLIENT
./files/usr/lib
./files/usr/lib/mozilla
./files/usr/lib/mozilla/CS_KOELN.tar.gz
./class
./class/dom.var
./class/NET_9.var
./mk-packages-gz
./scripts
./scripts/kueppers
./README
./disk_config
./disk_config/dom
./disk_config/kueppers
```

Le fichier **mk-packages-gz** est juste un simple script qui crée le **Packages.gz** comme expliqué ci-dessus. Dans le but de copier ces données de configuration locales dans l'espace de configuration *fai*, nous utilisons cette commande :

```
cp -a ~admin/additional-fai/* /usr/local/share/fai
```

Si vous supprimez un fichier dans votre configuration locale, n'oubliez pas de retirer ce fichier aussi dans

l'espace de configuration, sinon il sera toujours utilisé.

Après l'appel de **set-disk-info** , une liste de tous les disques durs locaux est enregistrée dans `$disklist` et `$device_size` contient une liste des disques et leurs tailles.

Utiliser **fai-divert** `-a` si un script de *postinstall* appelle un programme de configuration, par exemple le script de *postinstall* pour le paquet *apache* appelle **apacheconfig** , qui a besoin d'une entrée manuelle. Vous pouvez truquer le programme de configuration, alors l'installation peut être entièrement automatique. Mais n'oubliez pas d'utiliser **fai-divert** `-R` pour enlever tout le faux script.

Pendant l'installation vous pouvez exécuter des commandes à l'intérieur du système nouvellement installé dans un environnement chrooté en utilisant **chroot** ***`/tmp/target`*** ou juste **$ROOTCMD** suivi par la commande que vous voulez appeler. Par exemple **$ROOTCMD dpkg -l** montre les paquets installés sur le nouveau système.

La seule tâche qui doit être faite manuellement pour le nouveau matériel est d'assigner l'adresse MAC à un *hostname* et à une adresse IP et de définir les classes pour ce client si les fichiers de configuration existants ne sont pas assez génériques pour traiter cette nouvelle machine.

Il y a une grande différence entre l'écriture de quelques grands scripts de configuration, ou beaucoup de scripts très courts, un pour chaque classe. De grands script peuvent distinguer des classes en utilisant des déclarations de cas, le test `ifclass` ou avec des mécanismes de classe pour des script cfengine.

Si votre ordinateur ne peut pas bouter sur la carte réseau, vous n'avez pas toujours besoin de bouter sur la disquette. Ajoutez la classe `FAI_BOOTPART` et FAI créera automatiquement une entrée **lilo** ou **grub** pour démarrer le FAI **bootfloppy** sur cette partition. Vous pouvez alors démarrer la réinstallation sans disquette de boot. Cela raccourcira aussi la phase de test, car le démarrage depuis le disque dur est beaucoup plus rapide que le démarrage sur la disquette. Vous pouvez aussi mettre un mot de passe pour ce menu de démarrage.

Fonctions utiles pour les administrateurs avancés

fai-divert
: Ajoute ou enlève un fichier de la liste de «diversions» et remplace le fichier par un script factice. C'est utile quand un script de postinstall a besoin d'une entrée manuelle. À la fin de l'installation toutes les diversions sont retirées.

skiptask
: permet d'indiquer la liste de tâches qui seront sautées. Vous trouverez un exemple dans `partition.DISKLESS`.